

O'REILLY®

6th Edition

# Oracle PL/SQL Programming

COVERS VERSIONS THROUGH ORACLE DATABASE 12C



Steven Feuerstein  
*with Bill Pribyl*

# Oracle PL/SQL Programming

Considered the best Oracle PL/SQL programming guide by the Oracle community, this definitive guide is precisely what you need to make the most of Oracle's powerful procedural language. The sixth edition describes the features and capabilities of PL/SQL up through Oracle Database 12c Release 1.

Hundreds of thousands of PL/SQL developers have benefitted from this book over the last twenty years; this edition continues that tradition. With extensive code examples and a lively sense of humor, this book explains language fundamentals, explores advanced coding techniques, and offers best practices to help you solve real-world problems.

- Get PL/SQL programs up and running quickly, with clear instructions for executing, tracing, testing, debugging, and managing code
- Learn new 12.1 features, including the ACCESSIBLE\_BY clause, WITH FUNCTION and UDF pragma, and BEQUEATH CURRENT\_USER for views
- Take advantage of extensive code samples, from easy-to-follow examples to reusable packaged utilities
- Optimize PL/SQL performance with features like the function result cache and Oracle utilities such as PL/Scope and the PL/SQL hierarchical profiler
- Build modular, easy-to-maintain PL/SQL applications using packages, procedures, functions, and triggers

---

**Steven Feuerstein** is considered one of the world's leading experts on the Oracle PL/SQL language. Steven's been writing about—and training developers on—PL/SQL since 1990. He's the author of ten books on the subject, including *Oracle PL/SQL Best Practices*.

**Bill Pribyl** is an author, teacher, and software consultant. He's the author of *Learning Oracle PL/SQL* and coauthor of *Oracle PL/SQL Programming* and *Oracle PL/SQL Language Pocket Reference*.

---

## DATABASES

US \$69.99      CAN \$73.99

ISBN: 978-1-449-32445-2



5 6 9 9 9



Twitter: @oreillymedia  
facebook.com/oreilly

SIXTH EDITION

---

# Oracle PL/SQL Programming

*Steven Feuerstein  
with Bill Pribyl*

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'REILLY®

## **Oracle PL/SQL Programming, Sixth Edition**

by Steven Feuerstein with Bill Pribyl

Copyright © 2014 Steven Feuerstein, Bill Pribyl. All rights reserved.

Printed in the United States of America.

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

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (<http://my.safaribooksonline.com>). For more information, contact our corporate/institutional sales department: 800-998-9938 or [corporate@oreilly.com](mailto:corporate@oreilly.com).

**Editor:** Ann Spencer

**Indexer:** Ellen Troutman

**Production Editor:** Nicole Shelby

**Cover Designer:** Randy Comer

**Copyeditor:** Rachel Monaghan

**Interior Designer:** David Futato

**Proofreader:** Rachel Head

**Illustrator:** Rebecca Demarest

September 1995: First Edition

September 1997: Second Edition

September 2002: Third Edition

August 2005: Fourth Edition

September 2009: Fifth Edition

January 2014: Sixth Edition

### **Revision History for the Sixth Edition:**

2014-01-22: First release

See <http://oreilly.com/catalog/errata.csp?isbn=9781449324452> for release details.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. *Oracle PL/SQL Programming*, the image of ants, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc., was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and authors assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 978-1-449-32445-2

[QG]

*To my wife, Veva Silva, whose intelligence, strength, beauty, and art  
have greatly enriched my life.*

—Steven Feuerstein

*To my wife, Norma. Still melting my heart after a quarter century.*

—Bill Pribyl



---

# Table of Contents

Preface.....	xxv
--------------	-----

---

## Part I. Programming in PL/SQL

<b>1. Introduction to PL/SQL.....</b>	<b>3</b>
What Is PL/SQL?	3
The Origins of PL/SQL	4
The Early Years of PL/SQL	4
Improved Application Portability	5
Improved Execution Authority and Transaction Integrity	5
Humble Beginnings, Steady Improvement	6
So This Is PL/SQL	7
Integration with SQL	7
Control and Conditional Logic	8
When Things Go Wrong	9
About PL/SQL Versions	11
Oracle Database 12c New PL/SQL Features	12
Resources for PL/SQL Developers	14
The O'Reilly PL/SQL Series	15
PL/SQL on the Internet	16
Some Words of Advice	17
Don't Be in Such a Hurry!	17
Don't Be Afraid to Ask for Help	18
Take a Creative, Even Radical Approach	19
<b>2. Creating and Running PL/SQL Code.....</b>	<b>21</b>
Navigating the Database	21
Creating and Editing Source Code	22
SQL*Plus	23

---

Starting Up SQL*Plus	24
Running a SQL Statement	26
Running a PL/SQL Program	27
Running a Script	29
What Is the “Current Directory”?	30
Other SQL*Plus Tasks	31
Error Handling in SQL*Plus	36
Why You Will Love and Hate SQL*Plus	36
Performing Essential PL/SQL Tasks	37
Creating a Stored Program	37
Executing a Stored Program	41
Showing Stored Programs	41
Managing Grants and Synonyms for Stored Programs	42
Dropping a Stored Program	43
Hiding the Source Code of a Stored Program	44
Editing Environments for PL/SQL	44
Calling PL/SQL from Other Languages	45
C: Using Oracle’s Precompiler (Pro*C)	46
Java: Using JDBC	47
Perl: Using Perl DBI and DBD::Oracle	48
PHP: Using Oracle Extensions	49
PL/SQL Server Pages	51
And Where Else?	51
<b>3. Language Fundamentals.....</b>	<b>53</b>
PL/SQL Block Structure	53
Anonymous Blocks	55
Named Blocks	57
Nested Blocks	57
Scope	58
Qualify All References to Variables and Columns in SQL Statements	59
Visibility	62
The PL/SQL Character Set	65
Identifiers	67
Reserved Words	68
Whitespace and Keywords	70
Literals	70
NULLs	71
Embedding Single Quotes Inside a Literal String	72
Numeric Literals	73
Boolean Literals	74
The Semicolon Delimiter	74

Comments	75
Single-Line Comment Syntax	75
Multiline Comment Syntax	76
The PRAGMA Keyword	76
Labels	77

---

## Part II. PL/SQL Program Structure

<b>4. Conditional and Sequential Control.....</b>	<b>83</b>
IF Statements	83
The IF-THEN Combination	84
The IF-THEN-ELSE Combination	86
The IF-THEN-ELSIF Combination	87
Avoiding IF Syntax Gotchas	89
Nested IF Statements	90
Short-Circuit Evaluation	91
CASE Statements and Expressions	93
Simple CASE Statements	93
Searched CASE Statements	95
Nested CASE Statements	98
CASE Expressions	98
The GOTO Statement	100
The NULL Statement	101
Improving Program Readability	101
Using NULL After a Label	102
<b>5. Iterative Processing with Loops.....</b>	<b>105</b>
Loop Basics	105
Examples of Different Loops	106
Structure of PL/SQL Loops	107
The Simple Loop	108
Terminating a Simple Loop: EXIT and EXIT WHEN	109
Emulating a REPEAT UNTIL Loop	110
The Intentionally Infinite Loop	111
The WHILE Loop	112
The Numeric FOR Loop	114
Rules for Numeric FOR Loops	114
Examples of Numeric FOR Loops	115
Handling Nontrivial Increments	116
The Cursor FOR Loop	117
Example of Cursor FOR Loops	118

Loop Labels	119
The CONTINUE Statement	120
Tips for Iterative Processing	123
Use Understandable Names for Loop Indexes	123
The Proper Way to Say Goodbye	124
Obtaining Information About FOR Loop Execution	126
SQL Statement as Loop	126
<b>6. Exception Handlers.....</b>	<b>129</b>
Exception-Handling Concepts and Terminology	129
Defining Exceptions	132
Declaring Named Exceptions	132
Associating Exception Names with Error Codes	133
About Named System Exceptions	136
Scope of an Exception	139
Raising Exceptions	140
The RAISE Statement	140
Using RAISE_APPLICATION_ERROR	141
Handling Exceptions	143
Built-in Error Functions	144
Combining Multiple Exceptions in a Single Handler	149
Unhandled Exceptions	149
Propagation of Unhandled Exceptions	150
Continuing Past Exceptions	153
Writing WHEN OTHERS Handling Code	155
Building an Effective Error Management Architecture	157
Decide on Your Error Management Strategy	158
Standardize Handling of Different Types of Exceptions	159
Organize Use of Application-Specific Error Codes	162
Use Standardized Error Management Programs	163
Work with Your Own Exception “Objects”	165
Create Standard Templates for Common Error Handling	167
Making the Most of PL/SQL Error Management	169

---

## Part III. PL/SQL Program Data

<b>7. Working with Program Data.....</b>	<b>173</b>
Naming Your Program Data	173
Overview of PL/SQL Datatypes	175
Character Data	176
Numbers	177

Dates, Timestamps, and Intervals	178
Booleans	178
Binary Data	179
ROWIDs	179
REF CURSORS	179
Internet Datatypes	180
“Any” Datatypes	180
User-Defined Datatypes	181
Declaring Program Data	181
Declaring a Variable	181
Declaring Constants	182
The NOT NULL Clause	183
Anchored Declarations	183
Anchoring to Cursors and Tables	185
Benefits of Anchored Declarations	186
Anchoring to NOT NULL Datatypes	188
Programmer-Defined Subtypes	188
Conversion Between Datatypes	189
Implicit Data Conversion	189
Explicit Datatype Conversion	192
<b>8. Strings.....</b>	<b>199</b>
String Datatypes	199
The VARCHAR2 Datatype	200
The CHAR Datatype	201
String Subtypes	202
Working with Strings	203
Specifying String Constants	203
Using Nonprintable Characters	205
Concatenating Strings	206
Dealing with Case	207
Traditional Searching, Extracting, and Replacing	210
Padding	213
Trimming	215
Regular Expression Searching, Extracting, and Replacing	216
Working with Empty Strings	227
Mixing CHAR and VARCHAR2 Values	229
String Function Quick Reference	231
<b>9. Numbers.....</b>	<b>241</b>
Numeric Datatypes	241
The NUMBER Type	242

The PLS_INTEGER Type	247
The BINARY_INTEGER Type	248
The SIMPLE_INTEGER Type	249
The BINARY_FLOAT and BINARY_DOUBLE Types	251
The SIMPLE_FLOAT and SIMPLE_DOUBLE Types	256
Numeric Subtypes	256
Number Conversions	257
The TO_NUMBER Function	258
The TO_CHAR Function	261
The CAST Function	267
Implicit Conversions	268
Numeric Operators	270
Numeric Functions	271
Rounding and Truncation Functions	271
Trigonometric Functions	272
Numeric Function Quick Reference	272
<b>10. Dates and Timestamps.....</b>	<b>277</b>
Datetime Datatypes	278
Declaring Datetime Variables	280
Choosing a Datetime Datatype	281
Getting the Current Date and Time	282
Interval Datatypes	284
Declaring INTERVAL Variables	286
When to Use INTERVALS	287
Datetime Conversions	289
From Strings to Datetimes	289
From Datetimes to Strings	292
Working with Time Zones	295
Requiring a Format Mask to Match Exactly	298
Easing Up on Exact Matches	299
Interpreting Two-Digit Years in a Sliding Window	299
Converting Time Zones to Character Strings	301
Padding Output with Fill Mode	302
Date and Timestamp Literals	302
Interval Conversions	304
Converting from Numbers to Intervals	304
Converting Strings to Intervals	305
Formatting Intervals for Display	306
Interval Literals	307
CAST and EXTRACT	308
The CAST Function	308

The EXTRACT Function	310
Datetime Arithmetic	311
Date Arithmetic with Intervals and Datetimes	311
Date Arithmetic with DATE Datatypes	312
Computing the Interval Between Two Datetimes	313
Mixing DATEs and TIMESTAMPs	316
Adding and Subtracting Intervals	317
Multiplying and Dividing Intervals	317
Using Unconstrained INTERVAL Types	318
Date/Time Function Quick Reference	319
<b>11. Records.....</b>	<b>323</b>
Records in PL/SQL	323
Benefits of Using Records	324
Declaring Records	326
Programmer-Defined Records	327
Working with Records	330
Comparing Records	337
Trigger Pseudorecords	338
<b>12. Collections.....</b>	<b>341</b>
Collections Overview	342
Collections Concepts and Terminology	343
Types of Collections	345
Collection Examples	345
Where You Can Use Collections	349
Choosing a Collection Type	354
Collection Methods (Built-ins)	356
The COUNT Method	357
The DELETE Method	358
The EXISTS Method	359
The EXTEND Method	360
The FIRST and LAST Methods	361
The LIMIT Method	362
The PRIOR and NEXT Methods	362
The TRIM Method	363
Working with Collections	365
Declaring Collection Types	365
Declaring and Initializing Collection Variables	369
Populating Collections with Data	374
Accessing Data Inside a Collection	379
Using String-Indexed Collections	380

Collections of Complex Datatypes	385
Multilevel Collections	389
Working with Collections in SQL	398
Nested Table Multiset Operations	406
Testing Equality and Membership of Nested Tables	408
Checking for Membership of an Element in a Nested Table	409
Performing High-Level Set Operations	409
Handling Duplicates in a Nested Table	411
Maintaining Schema-Level Collections	412
Necessary Privileges	412
Collections and the Data Dictionary	413
<b>13. Miscellaneous Datatypes.....</b>	<b>415</b>
The BOOLEAN Datatype	415
The RAW Datatype	417
The UROWID and ROWID Datatypes	417
Getting ROWIDs	418
Using ROWIDs	419
The LOB Datatypes	420
Working with LOBs	422
Understanding LOB Locators	423
Empty Versus NULL LOBs	425
Writing into a LOB	427
Reading from a LOB	430
BFILEs Are Different	431
SecureFiles Versus BasicFiles	436
Temporary LOBs	439
Native LOB Operations	442
LOB Conversion Functions	447
Predefined Object Types	447
The XMLType Type	448
The URI Types	451
The Any Types	453

---

## Part IV. SQL in PL/SQL

<b>14. DML and Transaction Management.....</b>	<b>461</b>
DML in PL/SQL	462
A Quick Introduction to DML	462
Cursor Attributes for DML Operations	466
RETURNING Information from DML Statements	467

DML and Exception Handling	468
DML and Records	470
Transaction Management	473
The COMMIT Statement	474
The ROLLBACK Statement	474
The SAVEPOINT Statement	475
The SET TRANSACTION Statement	476
The LOCK TABLE Statement	476
Autonomous Transactions	477
Defining Autonomous Transactions	478
Rules and Restrictions on Autonomous Transactions	479
Transaction Visibility	480
When to Use Autonomous Transactions	481
Building an Autonomous Logging Mechanism	482
<b>15. Data Retrieval.....</b>	<b>485</b>
Cursor Basics	486
Some Data Retrieval Terms	487
Typical Query Operations	488
Introduction to Cursor Attributes	489
Referencing PL/SQL Variables in a Cursor	492
Choosing Between Explicit and Implicit Cursors	493
Working with Implicit Cursors	494
Implicit Cursor Examples	495
Error Handling with Implicit Cursors	496
Implicit SQL Cursor Attributes	498
Working with Explicit Cursors	500
Declaring Explicit Cursors	501
Opening Explicit Cursors	504
Fetching from Explicit Cursors	505
Column Aliases in Explicit Cursors	507
Closing Explicit Cursors	508
Explicit Cursor Attributes	510
Cursor Parameters	512
SELECT...FOR UPDATE	515
Releasing Locks with COMMIT	516
The WHERE CURRENT OF Clause	518
Cursor Variables and REF CURSORs	519
Why Use Cursor Variables?	520
Similarities to Static Cursors	521
Declaring REF CURSOR Types	521
Declaring Cursor Variables	522

Opening Cursor Variables	523
Fetching from Cursor Variables	524
Rules for Cursor Variables	527
Passing Cursor Variables as Arguments	530
Cursor Variable Restrictions	532
Cursor Expressions	533
Using Cursor Expressions	534
Restrictions on Cursor Expressions	536
<b>16. Dynamic SQL and Dynamic PL/SQL.....</b>	<b>537</b>
NDS Statements	538
The EXECUTE IMMEDIATE Statement	538
The OPEN FOR Statement	543
About the Four Dynamic SQL Methods	548
Binding Variables	550
Argument Modes	551
Duplicate Placeholders	553
Passing NULL Values	554
Working with Objects and Collections	555
Dynamic PL/SQL	557
Build Dynamic PL/SQL Blocks	558
Replace Repetitive Code with Dynamic Blocks	560
Recommendations for NDS	561
Use Invoker Rights for Shared Programs	561
Anticipate and Handle Dynamic Errors	562
Use Binding Rather than Concatenation	564
Minimize the Dangers of Code Injection	566
When to Use DBMS_SQL	569
Obtain Information About Query Columns	569
Meeting Method 4 Dynamic SQL Requirements	571
Minimizing Parsing of Dynamic Cursors	578
Oracle Database 11g New Dynamic SQL Features	579
Enhanced Security for DBMS_SQL	584

---

## Part V. PL/SQL Application Construction

<b>17. Procedures, Functions, and Parameters.....</b>	<b>591</b>
Modular Code	592
Procedures	593
Calling a Procedure	596
The Procedure Header	596

The Procedure Body	596
The END Label	597
The RETURN Statement	597
Functions	597
Structure of a Function	598
The RETURN Datatype	601
The END Label	602
Calling a Function	603
Functions Without Parameters	604
The Function Header	604
The Function Body	605
The RETURN Statement	605
Parameters	607
Defining Parameters	608
Actual and Formal Parameters	608
Parameter Modes	609
Explicit Association of Actual and Formal Parameters in PL/SQL	613
The NOCOPY Parameter Mode Qualifier	617
Default Values	618
Local or Nested Modules	619
Benefits of Local Modularization	620
Scope of Local Modules	623
Sprucing Up Your Code with Nested Subprograms	623
Subprogram Overloading	624
Benefits of Overloading	625
Restrictions on Overloading	628
Overloading with Numeric Types	629
Forward Declarations	630
Advanced Topics	631
Calling Your Function from Inside SQL	631
Table Functions	637
Deterministic Functions	647
Implicit Cursor Results (Oracle Database 12c)	649
Go Forth and Modularize!	650
<b>18. Packages.....</b>	<b>651</b>
Why Packages?	651
Demonstrating the Power of the Package	652
Some Package-Related Concepts	655
Diagramming Privacy	657
Rules for Building Packages	658
The Package Specification	658

The Package Body	660
Initializing Packages	662
Rules for Calling Packaged Elements	666
Working with Package Data	667
Global Within a Single Oracle Session	668
Global Public Data	669
Packaged Cursors	669
Serializable Packages	674
When to Use Packages	677
Encapsulate Data Access	677
Avoid Hardcoding Literals	680
Improve Usability of Built-in Features	683
Group Together Logically Related Functionality	683
Cache Static Session Data	684
Packages and Object Types	685
<b>19. Triggers.....</b>	<b>687</b>
DML Triggers	688
DML Trigger Concepts	689
Creating a DML Trigger	691
DML Trigger Example: No Cheating Allowed!	696
Multiple Triggers of the Same Type	702
Who Follows Whom	703
Mutating Table Errors	705
Compound Triggers: Putting It All in One Place	706
DDL Triggers	710
Creating a DDL Trigger	710
Available Events	713
Available Attributes	713
Working with Events and Attributes	715
Dropping the Undroppable	718
The INSTEAD OF CREATE Trigger	719
Database Event Triggers	720
Creating a Database Event Trigger	721
The STARTUP Trigger	722
The SHUTDOWN Trigger	723
The LOGON Trigger	723
The LOGOFF Trigger	723
The SERVERERROR Trigger	724
INSTEAD OF Triggers	728
Creating an INSTEAD OF Trigger	728
The INSTEAD OF INSERT Trigger	730

The INSTEAD OF UPDATE Trigger	732
The INSTEAD OF DELETE Trigger	733
Populating the Tables	733
INSTEAD OF Triggers on Nested Tables	734
AFTER SUSPEND Triggers	736
Setting Up for the AFTER SUSPEND Trigger	736
Looking at the Actual Trigger	738
The ORA_SPACE_ERROR_INFO Function	739
The DBMS_RESUMABLE Package	740
Trapped Multiple Times	742
To Fix or Not to Fix?	743
Maintaining Triggers	743
Disabling, Enabling, and Dropping Triggers	743
Creating Disabled Triggers	744
Viewing Triggers	745
Checking the Validity of Triggers	746
<b>20. Managing PL/SQL Code.....</b>	<b>749</b>
Managing Code in the Database	750
Overview of Data Dictionary Views	751
Display Information About Stored Objects	753
Display and Search Source Code	753
Use Program Size to Determine Pinning Requirements	755
Obtain Properties of Stored Code	756
Analyze and Modify Trigger State Through Views	757
Analyze Argument Information	758
Analyze Identifier Usage (Oracle Database 11g's PL/Scope)	759
Managing Dependencies and Recompiling Code	762
Analyzing Dependencies with Data Dictionary Views	763
Fine-Grained Dependency (Oracle Database 11g)	767
Remote Dependencies	769
Limitations of Oracle's Remote Invocation Model	772
Recompiling Invalid Program Units	773
Compile-Time Warnings	777
A Quick Example	777
Enabling Compile-Time Warnings	778
Some Handy Warnings	780
Testing PL/SQL Programs	788
Typical, Tawdry Testing Techniques	789
General Advice for Testing PL/SQL Code	793
Automated Testing Options for PL/SQL	794
Tracing PL/SQL Execution	795

DBMS_UTILITY.FORMAT_CALL_STACK	796
UTL_CALL_STACK (Oracle Database 12c)	798
DBMS_APPLICATION_INFO	801
Tracing with <code>opp_trace</code>	803
The DBMS_TRACE Facility	804
Debugging PL/SQL Programs	808
The Wrong Way to Debug	809
Debugging Tips and Strategies	811
Using Whitelisting to Control Access to Program Units	816
Protecting Stored Code	818
Restrictions on and Limitations of Wrapping	818
Using the Wrap Executable	819
Dynamic Wrapping with DBMS_DDL	819
Guidelines for Working with Wrapped Code	821
Introduction to Edition-Based Redefinition (Oracle Database 11g Release 2)	821
<b>21. Optimizing PL/SQL Performance.....</b>	<b>825</b>
Tools to Assist in Optimization	827
Analyzing Memory Usage	827
Identifying Bottlenecks in PL/SQL Code	827
Calculating Elapsed Time	833
Choosing the Fastest Program	834
Avoiding Infinite Loops	836
Performance-Related Warnings	837
The Optimizing Compiler	838
Insights on How the Optimizer Works	840
Runtime Optimization of Fetch Loops	843
Data Caching Techniques	844
Package-Based Caching	845
Deterministic Function Caching	850
THe Function Result Cache (Oracle Database 11g)	852
Caching Summary	868
Bulk Processing for Repeated SQL Statement Execution	869
High-Speed Querying with BULK COLLECT	870
High-Speed DML with FORALL	877
Improving Performance with Pipelined Table Functions	888
Replacing Row-Based Inserts with Pipelined Function-Based Loads	889
Tuning Merge Operations with Pipelined Functions	896
Asynchronous Data Unloading with Parallel Pipelined Functions	898
Performance Implications of Partitioning and Streaming Clauses in Parallel Pipelined Functions	902
Pipelined Functions and the Cost-Based Optimizer	903

Tuning Complex Data Loads with Pipelined Functions	909
A Final Word on Pipelined Functions	916
Specialized Optimization Techniques	917
Using the NOCOPY Parameter Mode Hint	917
Using the Right Datatype	921
Optimizing Function Performance in SQL (12.1 and higher)	922
Stepping Back for the Big Picture on Performance	923
<b>22. I/O and PL/SQL.....</b>	<b>925</b>
Displaying Information	925
Enabling DBMS_OUTPUT	926
Write Lines to the Buffer	926
Read the Contents of the Buffer	927
Reading and Writing Files	929
The UTL_FILE_DIR Parameter	929
Working with Oracle Directories	931
Open Files	932
Is the File Already Open?	934
Close Files	934
Read from Files	935
Write to Files	938
Copy Files	941
Delete Files	942
Rename and Move Files	943
Retrieve File Attributes	943
Sending Email	944
Oracle Prerequisites	945
Configuring Network Security	946
Send a Short (32,767 Bytes or Less) Plain-Text Message	947
Include “Friendly” Names in Email Addresses	948
Send a Plain-Text Message of Arbitrary Length	950
Send a Message with a Short (32,767 Bytes or Less) Attachment	951
Send a Small File (32,767 Bytes or Less) as an Attachment	953
Attach a File of Arbitrary Size	953
Working with Web-Based Data (HTTP)	956
Retrieve a Web Page in “Pieces”	956
Retrieve a Web Page into a LOB	958
Authenticate Using HTTP Username/Password	959
Retrieve an SSL-Encrypted Web Page (via HTTPS)	960
Submit Data to a Web Page via GET or POST	961
Disable Cookies or Make Cookies Persistent	965
Retrieve Data from an FTP Server	966

Use a Proxy Server	966
Other Types of I/O Available in PL/SQL	967
Database Pipes, Queues, and Alerts	967
TCP Sockets	968
Oracle's Built-in Web Server	968

---

## Part VI. Advanced PL/SQL Topics

<b>23. Application Security and PL/SQL.....</b>	<b>971</b>
Security Overview	971
Encryption	973
Key Length	974
Algorithms	975
Padding and Chaining	977
The DBMS_CRYPTO Package	977
Encrypting Data	979
Encrypting LOBs	982
SecureFiles	982
Decrypting Data	983
Performing Key Generation	984
Performing Key Management	985
Cryptographic Hashing	991
Using Message Authentication Codes	993
Using Transparent Data Encryption	994
Transparent Tablespace Encryption	997
Row-Level Security	999
Why Learn About RLS?	1002
A Simple RLS Example	1003
Static Versus Dynamic Policies	1007
Using Column-Sensitive RLS	1012
RLS Debugging	1015
Application Contexts	1019
Using Application Contexts	1020
Security in Contexts	1022
Contexts as Predicates in RLS	1022
Identifying Nondatabase Users	1026
Fine-Grained Auditing	1028
Why Learn About FGA?	1029
A Simple FGA Example	1030
Access How Many Columns?	1032
Checking the Audit Trail	1033

Using Bind Variables	1035
Using Handler Modules	1036
<b>24. PL/SQL Architecture.....</b>	<b>1039</b>
DIANA	1039
How Oracle Executes PL/SQL Code	1040
An Example	1041
Compiler Limits	1044
The Default Packages of PL/SQL	1045
Execution Authority Models	1048
The Definer Rights Model	1049
The Invoker Rights Model	1054
Combining Rights Models	1056
Granting Roles to PL/SQL Program Units (Oracle Database 12c)	1057
“Who Invoked Me?” Functions (Oracle Database 12c)	1060
BEQUEATH CURRENT_USER for Views (Oracle Database 12c)	1061
Constraining Invoker Rights Privileges (Oracle Database 12c)	1063
Conditional Compilation	1064
Examples of Conditional Compilation	1065
The Inquiry Directive	1066
The \$IF Directive	1070
The \$ERROR Directive	1072
Synchronizing Code with Packaged Constants	1072
Program-Specific Settings with Inquiry Directives	1073
Working with Postprocessed Code	1074
PL/SQL and Database Instance Memory	1076
The SGA, PGA, and UGA	1076
Cursors, Memory, and More	1077
Tips on Reducing Memory Use	1079
What to Do If You Run Out of Memory	1090
Native Compilation	1093
When to Run in Interpreted Mode	1094
When to Go Native	1094
Native Compilation and Database Release	1094
What You Need to Know	1095
<b>25. Globalization and Localization in PL/SQL.....</b>	<b>1097</b>
Overview and Terminology	1099
Unicode Primer	1100
National Character Set Datatypes	1102
Character Encoding	1102
Globalization Support Parameters	1104

Unicode Functions	1105
Character Semantics	1111
String Sort Order	1115
Binary Sort	1116
Monolingual Sort	1117
Multilingual Sort	1119
Multilingual Information Retrieval	1120
IR and PL/SQL	1123
Date/Time	1126
Timestamp Datatypes	1126
Date/Time Formatting	1127
Currency Conversion	1131
Globalization Development Kit for PL/SQL	1133
UTL_118N Utility Package	1133
UTL_LMS Error-Handling Package	1136
GDK Implementation Options	1137
<b>26. Object-Oriented Aspects of PL/SQL.....</b>	<b>1141</b>
Introduction to Oracle's Object Features	1142
Object Types by Example	1144
Creating a Base Type	1144
Creating a Subtype	1146
Methods	1147
Invoking Supertype Methods in Oracle Database 11g and Later	1152
Storing, Retrieving, and Using Persistent Objects	1154
Evolution and Creation	1162
Back to Pointers?	1164
Generic Data: The ANY Types	1171
I Can Do It Myself	1176
Comparing Objects	1179
Object Views	1184
A Sample Relational System	1186
Object View with a Collection Attribute	1188
Object Subview	1191
Object View with Inverse Relationship	1192
INSTEAD OF Triggers	1193
Differences Between Object Views and Object Tables	1196
Maintaining Object Types and Object Views	1197
Data Dictionary	1197
Privileges	1199

Concluding Thoughts from a (Mostly) Relational Developer	1201
<b>27. Calling Java from PL/SQL.....</b>	<b>1205</b>
Oracle and Java	1205
Getting Ready to Use Java in Oracle	1207
Installing Java	1207
Building and Compiling Your Java Code	1208
Setting Permissions for Java Development and Execution	1209
A Simple Demonstration	1212
Finding the Java Functionality	1212
Building a Custom Java Class	1213
Compiling and Loading into Oracle	1215
Building a PL/SQL Wrapper	1217
Deleting Files from PL/SQL	1217
Using loadjava	1218
Using dropjava	1221
Managing Java in the Database	1221
The Java Namespace in Oracle	1221
Examining Loaded Java Elements	1222
Using DBMS_JAVA	1223
LONGNAME: Converting Java Long Names	1223
GET_-, SET_-, and RESET_COMPILER_OPTION: Getting and Setting (a Few) Compiler Options	1224
SET_OUTPUT: Enabling Output from Java	1225
EXPORT_SOURCE, EXPORT_RESOURCE, and EXPORT_CLASS: Exporting Schema Objects	1226
Publishing and Using Java in PL/SQL	1228
Call Specs	1228
Some Rules for Call Specs	1229
Mapping Datatypes	1230
Calling a Java Method in SQL	1232
Exception Handling with Java	1232
Extending File I/O Capabilities	1236
Other Examples	1240
<b>28. External Procedures.....</b>	<b>1243</b>
Introduction to External Procedures	1244
Example: Invoking an Operating System Command	1244
Architecture of External Procedures	1246
Oracle Net Configuration	1248
Specifying the Listener Configuration	1248
Security Characteristics of the Configuration	1251

Setting Up Multithreaded Mode	1252
Creating an Oracle Library	1254
Writing the Call Specification	1256
The Call Spec: Overall Syntax	1257
Parameter Mapping: The Example Revisited	1258
Parameter Mapping: The Full Story	1260
More Syntax: The PARAMETERS Clause	1262
PARAMETERS Properties	1263
Raising an Exception from the Called C Program	1266
Nondefault Agents	1269
Maintaining External Procedures	1272
Dropping Libraries	1272
Data Dictionary	1272
Rules and Warnings	1273
<b>A. Regular Expression Metacharacters and Function Parameters.</b>	<b>1275</b>
<b>B. Number Format Models.</b>	<b>1281</b>
<b>C. Date Format Models.</b>	<b>1285</b>
<b>Index.</b>	<b>1291</b>

---

# Preface

Millions of application developers and database administrators around the world use software provided by Oracle Corporation to build complex systems that manage vast quantities of data. At the heart of much of Oracle's software is PL/SQL—a programming language that provides procedural extensions to Oracle's version of SQL (Structured Query Language) and serves as the programming language within the Oracle Developer toolset (most notably Forms Developer and Reports Developer).

PL/SQL figures prominently as an enabling technology in almost every new product released by Oracle Corporation. Software professionals use PL/SQL to perform many kinds of programming functions, including:

- Implementing crucial business rules in the Oracle Server with PL/SQL-based stored procedures and database triggers
- Generating and managing XML documents entirely within the database
- Linking web pages to an Oracle database
- Implementing and automating database administration tasks—from establishing row-level security to managing rollback segments within PL/SQL programs

PL/SQL was modeled after Ada,<sup>1</sup> a programming language designed for the US Department of Defense. Ada is a high-level language that emphasizes data abstraction, information hiding, and other key elements of modern design strategies. As a result of this very smart design decision by Oracle, PL/SQL is a powerful language that incorporates many of the most advanced elements of procedural languages, including:

- A full range of datatypes from number to string, and including complex data structures such as records (which are similar to rows in a relational table), collections

---

1. The language was named “Ada” in honor of Ada Lovelace, a mathematician who is regarded by many to have been the world’s first computer programmer.

(which are Oracle's version of arrays), and XMLType (for managing XML documents in Oracle and through PL/SQL)

- An explicit and highly readable block structure that makes it easy to enhance and maintain PL/SQL applications
- Conditional, iterative, and sequential control statements, including a CASE statement and three different kinds of loops
- Exception handlers for use in event-based error handling
- Named, reusable code elements such as functions, procedures, triggers, object types (akin to object-oriented classes), and packages (collections of related programs and variables)

PL/SQL is integrated tightly into Oracle's SQL language: you can execute SQL statements directly from your procedural program without having to rely on any kind of intermediate application programming interface (API) such as Java Database Connectivity (JDBC) or Open Database Connectivity (ODBC). Conversely, you can also call your own PL/SQL functions from within a SQL statement.

Oracle developers who want to be successful in the 21st century must learn to use PL/SQL to full advantage. This is a two-step process. First, you must become familiar with and learn how to use the language's ever-expanding set of features; and second, after gaining competence in the individual features, you must learn how to put these constructs together to build complex applications.

For these reasons and more, Oracle developers need a solid, comprehensive resource for the base PL/SQL language. You need to know the basic building blocks of PL/SQL, but you also need to learn by example so that you can avoid some of the trial and error. As with any programming language, PL/SQL has a right way and many wrong ways (or at least "not as right" ways) to handle just about any task. It is my hope that this book will help you learn how to use the PL/SQL language in the most effective and efficient way possible.

## Objectives of This Book

What, specifically, will this book help you do?

### *Take full advantage of PL/SQL*

Oracle's reference manuals may describe all the features of the PL/SQL language, but they don't tell you how to apply the technology. In fact, in some cases, you'll be lucky to even understand how to use a given feature after you've made your way through the railroad diagrams. Books and training courses tend to cover the same standard topics in the same limited way. In this book, I'll venture beyond the basics to the far reaches of the language, finding the nonstandard ways that a particular feature can be tweaked to achieve a desired result.

### *Use PL/SQL to solve your problems*

You don't spend your days and nights writing PL/SQL modules so that you can rise to a higher plane of existence. You use PL/SQL to solve problems for your company or your customers. In this book, I try hard to help you tackle real-world problems, the kinds of issues developers face on a daily basis (at least those problems that can be solved with mere software). To do this, I've packed the book with examples—not just small code fragments, but substantial application components that you can apply immediately to your own situations. There is a good deal of code in the book itself, and much more on the accompanying website. In a number of cases, I use the code examples to guide you through the analytical process needed to come up with a solution. In this way you'll see, in the most concrete terms, how to apply PL/SQL features and undocumented applications of those features to a particular situation.

### *Write efficient, maintainable code*

PL/SQL and the rest of the Oracle products offer the potential for incredible development productivity. If you aren't careful, however, this capability will simply let you dig yourself into a deeper, darker hole than you've ever found yourself in before. I would consider this book a failure if it only helped programmers write more code in less time; I want to help you develop the skills and techniques to build applications that readily adapt to change and that are easily understood and maintained. I want to teach you to use comprehensive strategies and code architectures that allow you to apply PL/SQL in powerful, general ways to the problems you face.

## Structure of This Book

Both the authors and O'Reilly Media are committed to providing comprehensive, useful coverage of PL/SQL over the life of the language. This sixth edition of *Oracle PL/SQL Programming* describes the features and capabilities of PL/SQL up through Oracle Database 12c Release 1. I assume for this edition that Oracle Database 12c is the baseline PL/SQL version. However, where appropriate, I reference specific features introduced (or only available) in other, earlier versions. For a list of the main characteristics of the various releases, see the section "[About PL/SQL Versions](#)" on page 11 in [Chapter 1](#).

PL/SQL has improved dramatically since the release of version 1.0 in the Oracle 6 database so many years ago. *Oracle PL/SQL Programming* has also undergone a series of major transformations to keep up with PL/SQL and provide ever-improving coverage of its features.

The biggest change in the sixth edition is its comprehensive coverage of all new PL/SQL features in Oracle Database 12c Release 1. The major features are summarized in [Chapter 1](#), along with references to the chapters where these features are discussed in detail.

I am very happy with the results and hope that you will be too. There is more information than ever before, but I think we managed to present it without losing the sense of humor and conversational tone that readers have told me for years make the book readable, understandable, and highly useful.

One comment regarding the “voice” behind the text. You may notice that in some parts of this book we use the word *we*, and in others *I*. One characteristic of this book (and one for which readers have expressed appreciation) is the personal voice that’s inseparable from the text. Consequently, even with the addition of coauthors to the book (and, in the third, fourth, and fifth editions, significant contributions from several other people), we’ve decided to maintain the use of *I* when an author speaks in his own voice.

Rather than leave you guessing as to which lead author is represented by the *I* in a given chapter, we thought we’d offer this quick guide for the curious; you’ll find additional discussion of our contributors in the Acknowledgments.

Chapter	Author	Chapter	Author
Preface	Steven	15	Steven
1	Steven	16	Steven
2	Bill and Steven	17	Steven
3	Steven and Bill	18	Steven
4	Steven, Chip, and Jonathan	19	Darryl and Steven
5	Steven and Bill	20	Steven
6	Steven	21	Steven and Adrian
7	Chip, Jonathan, and Steven	22	Bill and Steven
8	Chip, Jonathan, and Steven	23	Arup
9	Chip, Jonathan, and Steven	24	Bill, Steven, and Chip
10	Chip, Jonathan, and Steven	25	Ron
11	Steven	26	Bill and Steven
12	Steven and Bill	27	Bill and Steven
13	Chip and Jonathan	28	Bill and Steven
14	Steven		

## About the Contents

The sixth edition of *Oracle PL/SQL Programming* is divided into six parts:

### *Part I*

I start from the very beginning in [Chapter 1](#): where did PL/SQL come from? What is it good for? I offer a very quick review of some of the main features of the PL/SQL language. [Chapter 2](#) is designed to help you get PL/SQL programs up and running as quickly as possible: it contains clear, straightforward instructions for executing PL/SQL code in SQL\*Plus and a few other common environments.

**Chapter 3** reviews fundamentals of the PL/SQL language: what makes a PL/SQL statement, an introduction to the block structure, how to write comments in PL/SQL, and so on.

#### *Part II*

**Chapter 4** through **Chapter 6** explore conditional (IF and CASE) and sequential (GOTO and NULL) control statements, loops and the CONTINUE statement, and exception handling in the PL/SQL language. This section of the book will teach you to construct blocks of code that correlate to the complex requirements of your applications.

#### *Part III*

Just about every program you write will manipulate data, and much of that data will be local to (defined in) your PL/SQL procedure or function. **Chapter 7** through **Chapter 13** concentrate on the various types of program data you can define in PL/SQL, such as numbers, strings, dates, timestamps, records, and collections. You will learn about the new datatypes introduced in Oracle Database 11g (SIMPLE\_INTEGER, SIMPLE\_FLOAT, and SIMPLE\_DOUBLE), as well as the many binary, date, and timestamp types introduced in other recent releases. These chapters also cover the various built-in functions provided by Oracle that allow you to manipulate and modify data.

#### *Part IV*

**Chapter 14** through **Chapter 16** address one of the central elements of PL/SQL code construction: the connection to the underlying database, which takes place through the SQL language. These chapters show you how to define transactions that update, insert, merge, and delete tables in the database; how to query information from the database for processing in a PL/SQL program; and how to execute SQL statements dynamically, using native dynamic SQL (NDS).

#### *Part V*

This is where it all comes together. You know about declaring and working with variables, and you're an expert in error handling and loop construction. Now, in **Chapter 17** through **Chapter 22**, you'll learn about the building blocks of applications, which include procedures, functions, packages, and triggers, and how to move information into and out of PL/SQL programs. **Chapter 20** discusses managing your PL/SQL code base, including testing and debugging programs and managing dependencies; it also provides an overview of the edition-based redefinition capability introduced in Oracle Database 11g Release 2. **Chapter 21** focuses on how you can use a variety of tools and techniques to get the best performance out of your PL/SQL programs. **Chapter 22** covers I/O techniques for PL/SQL, from DBMS\_OUTPUT (writing output to the screen) and UTL\_FILE (reading and writing files) to UTL\_MAIL (sending mail) and UTL\_HTTP (retrieving data from a web page).

## *Part VI*

A language as mature and rich as PL/SQL is full of features that you may not use on a day-to-day basis, but that may make the crucial difference between success and failure. Chapter 23 explores the security-related challenges we face as we build PL/SQL programs. Chapter 24 contains an exploration of the PL/SQL architecture, including PL/SQL's use of memory. Chapter 25 provides guidance for PL/SQL developers who need to address issues of globalization and localization. Chapter 26 offers a guide to the object-oriented features of Oracle (object types and object views).

Appendix A through Appendix C summarize the details of regular expression syntax and number and date formats.

The chapters on invoking Java and C code from PL/SQL applications, which were part of the hardcopy fourth edition, have been moved to the book's website.

If you are new to PL/SQL, reading this book from beginning to end should improve your PL/SQL skills and deepen your understanding of the language. If you're already a proficient PL/SQL programmer, you'll probably want to dip into the appropriate sections to extract particular techniques for immediate application. Whether you use this book as a teaching guide or as a reference, I hope it will help you use PL/SQL effectively.

## **What This Book Does Not Cover**

As long as this book is, it doesn't contain everything. The Oracle environment is huge and complex, and in this book we've focused our attention on the core PL/SQL language itself. The following topics are therefore outside the scope of this book and are not covered, except in an occasional and peripheral fashion:

### *The SQL language*

I assume that you already have a working knowledge of the SQL language, and that you know how to write SELECTs, UPDATEs, INSERTs, MERGEs, and DELETEs.

### *Administration of Oracle databases*

While database administrators (DBAs) can use this book to learn how to write the PL/SQL needed to build and maintain databases, this book does not explore all the nuances of the Data Definition Language (DDL) of Oracle's SQL.

### *Application and database tuning*

I don't cover detailed tuning issues in this book, although Chapter 21 does discuss the many tools and techniques that will help you to optimize the performance of your PL/SQL programs.

### *Oracle tool-specific technologies independent of PL/SQL*

This book does not attempt to show you how to build applications in a tool like Oracle's Forms Developer, even though the implementation language is PL/SQL. I

have chosen to focus on core language capabilities, centered on what you can do with PL/SQL from within the database. However, almost everything covered in this book is applicable to PL/SQL inside Forms Developer and Reports Developer.

## Conventions Used in This Book

The following conventions are used in this book:

### *Italic*

Used for file and directory names and for emphasis when introducing a new term. In the text, it is also used to indicate a user-replaceable element.

### **Constant width**

Used for code examples.

### **Constant width bold**

Indicates user input in examples showing an interaction. Also, in some code examples, highlights the statements being discussed.

### *Constant width italic*

In some code examples, indicates an element (e.g., a parameter) that you supply.

### UPPERCASE

In code examples, generally indicates PL/SQL keywords or certain identifiers used by Oracle Corporation as built-in function and package names.

### lowercase

In code examples, generally indicates user-defined items such as variables, parameters, etc.

### Punctuation

In code examples, enter exactly as shown.

### Indentation

In code examples, helps to show structure but is not required.

--

In code examples, a double hyphen begins a single-line comment that extends to the end of a line.

### /\* and \*/

In code examples, these characters delimit a multiline comment that can extend from one line to another.

.

In code examples and related discussions, a dot qualifies a reference by separating an object name from a component name. For example, dot notation is used to select fields in a record and to specify declarations within a package.

[ ]

In syntax descriptions, square brackets enclose optional items.

{ }

In syntax descriptions, curly brackets enclose a set of items from which you must choose only one.

|

In syntax descriptions, a vertical bar separates the items enclosed in curly brackets, as in {TRUE | FALSE}.

...

In syntax descriptions, ellipses indicate repeating elements. An ellipsis also shows that statements or clauses irrelevant to the discussion were left out.



Indicates a tip, suggestion, or general note. For example, I'll tell you if a certain setting is version specific.



Indicates a warning or caution. For example, I'll tell you if a certain setting has some kind of negative impact on the system.

## Which Platform or Version?

In general, all the discussions and examples in this book apply regardless of the machine and/or operating system you are using. In those cases in which a feature is in any way version-dependent—for example, if you can use it only in Oracle Database 11g (or in a specific release, such as Oracle Database 11g Release 2)—I note that in the text.

There are many versions of PL/SQL, and you may find that you need to use multiple versions in your development work. [Chapter 1](#) describes the various versions of PL/SQL and what you should know about them; see “[About PL/SQL Versions](#)” on page 11.

## About the Code

All of the code referenced in this book is available from <http://oreil.ly/oracle-plsql-sixth>. You will also find the contents of some of the chapters from earlier editions that we removed or condensed in the different editions of the book. These may be especially helpful to readers who are running older versions of Oracle.

Information about all of Steven’s books and accompanying resources can be found at <http://www.stevenfeuerstein.com>. You might also want to visit PL/SQL Obsession (Steven Feuerstein’s PL/SQL portal) at [http://www.plsql.org](#), where you will find training materials, code downloads, and more.

To find a particular example on the book’s website, look for the filename cited in the text. For many examples, you will find filenames in the following form provided as a comment at the beginning of the example shown in the book, as illustrated here:

```
/* File on web: fullname.pkg */
```

If the code snippet in which you are interested does not have a “File on web” comment, then you should check the corresponding chapter code file.

A chapter code file contains all the code fragments and examples that do not merit their own file, but may prove useful to you for copy-and-paste operations. These files also contain the DDL statements to create tables and other objects on which the code may depend.

Each chapter code file is named *chNN\_code.sql*, where *NN* is the number of the chapter.

Finally, the *hr\_schema\_install.sql* script will create the standard Oracle Human Resources demonstration tables, such as employees and departments. These tables are used in examples throughout the book.

## Using Code Examples

Supplemental material (code examples, exercises, etc.) is available for download at <http://oreil.ly/oracle-plsql-sixth>.

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: “*Oracle PL/SQL Programming, Sixth Edition* by Steven Feuerstein and Bill Pribyl (O’Reilly). Copyright 2014 Steven Feuerstein and Bill Pribyl, 978-1-4493-2445-2.”

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](mailto:permissions@oreilly.com).

# Safari® Books Online



Safari Books Online ([www.safaribooksonline.com](http://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://oreil.ly/oracle-plsql-sixth>.

To comment or ask technical questions about this book, send email to [bookquestions@oreilly.com](mailto: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

Since *Oracle PL/SQL Programming* was first published in 1995, it has had a busy and productive history as the “go to” text on how to use the PL/SQL language. For that, I first of all express our appreciation to all our readers.

Maintaining *Oracle PL/SQL Programming* as an accurate, readable, and up-to-date reference to PL/SQL has been, from the start, a big (all right, I admit it—sometimes overwhelming) job; it certainly would not have been possible without the help of many Oracle specialists, friends, and family, and of course the incredible staff at O'Reilly Media.

You will find below rather detailed thank yous for those who helped pull together the sixth edition of *Oracle PL/SQL Programming*. Following that, you will find an acknowledgment of the many people who were instrumental in the earlier editions.

First and foremost, I thank those who contributed chapters and/or substantial content for the book; listed alphabetically, they are Adrian Billington, Chip Dawes, Jonathan Gennick, Ron Hardman, Darryl Hurley, and Arup Nanda. As of this edition, Chip Dawes has taken over responsibility for updating a half-dozen chapters. Jonathan Gennick wrote or substantially updated six chapters in past editions. Darryl Hurley has updated the fine chapter on database triggers for several editions and contributed insights on Oracle's internationalization features. Arup Nanda wrote the excellent chapter on security. Ron Hardman stepped up to the plate and wrote the chapter on globalization and localization. Adrian Billington provided excellent material in [Chapter 21](#) on pipelined table functions.

I have invited each of our contributors to say a few words about themselves.

**Adrian Billington** is a consultant in database design, development, and performance tuning who has been working with Oracle databases since 1999. He is the man behind [oracle-developer.net](#), a website full of SQL and PL/SQL features, utilities, and techniques for Oracle developers. Adrian is also an Oracle ACE and a member of the OakTable Network. He would like to thank James Padfield (Padders), Tom Kyte, and Steven Feuerstein for inspiring him to become a better developer during his impressionable early years as an Oracle professional. He lives in the UK with his wife Anji and three children, Georgia, Oliver, and Isabella.

**Chip Dawes** has been working with Oracle database technologies for over 20 years as a DBA, developer, teacher, and mentor. He is currently a manager at [PwC](#), where he helps clients find value in their data. Chip lives in Chicagoland with his wife and children.

**Jonathan Gennick** is an experienced technology professional who is well known for his Oracle database expertise. His past experience encompasses both software development and database administration. As a developer, he has always enjoyed troubleshooting

and debugging. He loves working with SQL and PL/SQL, and is well known for his books and articles on those topics. In his off hours, Jonathan enjoys a rather low-tech approach to life. He serves actively in his local church, where you'll often find him engaged in scripture with a class of high school and sometimes college-age students, or even speaking from the pulpit. He is also an avid mountain biker, riding even in the dead of winter on very cool, studded bicycle tires imported from Finland. In his Oracle work, he is currently working his way through an exploration of Oracle SQL's built-in statistic functions.

**Ron Hardman** is founder of [SettleOurEstate.com](#), an estate management solution built on Oracle Apex and the Oracle Cloud Database. He also consults around the world on Oracle Text and Oracle globalization technologies, and has been working with Oracle both as an employee and as a customer for more than 17 years. Ron enjoys writing about more than technology, releasing in 2010 his first historical fiction book, titled *Shadow Fox: Sons of Liberty*, which he cowrote with his daughter.

**Darryl Hurley** has been working with Oracle technology for more than 20 years, focusing on PL/SQL and DBA work. He lives in Richmond, British Columbia, with his lovely wife, Vanessa, and beautiful daughter, Bianca.

**Arup Nanda** has been an Oracle DBA since 1993, touching all aspects of the job—modeling, performance troubleshooting, PL/SQL coding, backups, disaster recovery, and more. He works as the principal database architect at a major corporation, has written about 500 articles, coauthored five books, and presented about 300 sessions at various conferences. He offers training sessions, engages in special projects like audits and DR, and writes about Oracle technology on his blog, [arup.blogspot.com](#). He was *Oracle Magazine*'s 2003 DBA of the Year and 2012 Architect of the Year. He is an OCP, an OTN ACE Director, and a member of the OakTable Network. He lives in Connecticut with his wife, Anu, and son, Anish.

With such a big book, we needed lots of reviewers, especially because we asked them to test each code snippet and program in the book to keep to an absolute minimum the number of errors that made it into the printed version. I am deeply grateful to the following men and women of the Oracle PL/SQL world, who took time away from the rest of their lives to help make *Oracle PL/SQL Programming* the best book that it could be.

For this sixth edition, I first thank Valentin Nikotin, one of the best technical reviewers I've ever had for this book. He not only checked the Oracle Database 12c content for accuracy, but also helped me remove ambiguities and correct mistakes in several other key chapters that had not changed for this new edition. My other technical reviewers also had a big impact on the quality of this book. Many thanks, Patrick Barel and Arup Nanda!

Next, I offer my deep appreciation to Bryn Llewellyn, Oracle’s PL/SQL Product Manager, and other members of the PL/SQL development team, most notably Charles Wetherell. Bryn provided crucial information and feedback on Oracle Database 12c’s new features and answered endless questions about various PL/SQL features with bottomless patience. There is no doubt that my understanding of PL/SQL and the accuracy with which I present it owe a great debt to Bryn.

From a non-Oracle perspective, grateful thoughts go to Joel Finkel, my favorite jack-of-all-trades, who makes up for the narrow specialization that simultaneously benefits and constrains *my* capabilities when it comes to computers and software.

Of course, that’s just the technical content. Once I feel that we’ve got our treatment of PL/SQL “right,” it’s time for the remarkable crew at O’Reilly Media—led by my editor, Ann Spencer—to transform our many chapters and code examples into a book worthy of the O’Reilly imprint. Many thanks to Julie Steele, editor of the fifth edition; Nicole Shelby, production editor for this edition; Rob Romano, who created the excellent figures; and the rest of the crew. This was the first time that Ann edited my book. For all previous editions (that is, from 1994 to 2007), I had the great honor and pleasure of working with Debby Russell. Thanks, Debby, for your many years of commitment to making the entire O’Reilly Media Oracle series a big success!

And here are the many people we thanked (and continue to be grateful to) for their contributions to the first five editions of this book: Sohaib Abassi, Steve Adams, Don Bales, Cailein Barclay, Patrick Barel, John Beresniewicz, Tom Berthoff, Sunil Bhargava, Jennifer Blair, Dick Bolz, Bryan Boulton, Per Brondum, Boris Burshteyn, Eric Camplin, Joe Celko, Gary Cernosek, Barry Chase, Geoff Chester, Ivan Chong, Dan Clamage, Gray Clossman, Avery Cohen, Robert A. G. Cook, John Cordell, Steve Cosner, Tony Crawford, Daniel Cronk, Ervan Darnell, Lex de Haan, Thomas Dunbar, Bill Dwight, Steve Ehrlich, Larry Elkins, Bruce Epstein, Joel Finkel, R. James Forsythe, Mike Gangler, Beverly Gibson, Steve Gillis, Eric Givler, Rick Greenwald, Radhakrishna Hari, Gerard Hartgers, Donald Herkimer, Steve Hilker, Bill Hinman, Gabriel Hoffman, Chandrasekharan Iyer, Ken Jacobs, Hakan Jakobsson, Giovanni Jaramillo, Dwayne King, Marcel Kratochvil, Thomas Kurian, Tom Kyte, Ben Lindsey, Peter Linsley, Vadim Loevski, Leo Lok, Debra Luik, James Mallory, Raj Mattamal, Andrew McIlwrick, Nimish Mehta, Ari Mozes, Steve Muench, Jeff Muller, Kannan Muthukkaruppan, Dan Norris, Alex Nuijten, James Padfield, Rakesh Patel, Karen Peiser, Fred Polizo, Dave Posner, Patrick Pribyl, Nancy Priest, Shirish Puranik, Chris Racicot, Sri Rajan, Mark Richter, Chris Rimmer, Alex Romankevich, Bert Scalzo, Pete Schaffer, Drew Smith, Scott Sowers, JT Thomas, David Thompson, Edward Van Hatten, Peter Vasterd, Andre Vergison, Mark Vilroox, Zona Walcott, Bill Watkins, Charles Wetherell, Edward Wiles, Daniel Wong, Solomon Yakobson, Ming Hui Yang, and Tony Ziembra.

My wife, Veva Silva, has supported me every step of the way through my career in the world of software, and I thank her deeply. My boys, Christopher Tavares Silva and Eli

Silva Feuerstein, have tolerated very well the diversion of my attention from them to PL/SQL (and when they were teenagers, positively delighted in this diversion). And finally I thank Chris and his lovely, smart, and creative wife, Lauren, for providing me with my first grandchild, Loey Lucille Silva.