# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreword</td>
<td>xv</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>xvii</td>
</tr>
<tr>
<td>About the Author</td>
<td>xviii</td>
</tr>
<tr>
<td>Introduction</td>
<td>xix</td>
</tr>
<tr>
<td>What This Book Covers</td>
<td>xxii</td>
</tr>
<tr>
<td>Who is This Book For?</td>
<td>xxiii</td>
</tr>
<tr>
<td>Why a Separate Book?</td>
<td>xxiv</td>
</tr>
</tbody>
</table>

### Chapter 1  Embedded Programming—An Oracle-Centric Approach  

1.1 Embedded Programming from an Oracle Perspective: The Primary Indicators and the Solution Life Cycle  
2

1.2 What's In and What's Not: Programming Languages, Platforms, and Solution Options Available for an Oracle-Based Solution  
3

1.2.1 Using .NET-Based Languages  
5

1.2.2 Using Java-Based Languages (Java Libraries, Java via JDBC, Java via SQLJ, Java via Java Stored Procedures)  
6

1.2.3 Using PHP-Based Languages (PHP-only, PHP and Ajax, PHP Web Pages)  
9

1.2.4 The Foundations: Architecture, Technologies, and Methodologies for Integration and Interaction with Oracle  
11

1.3 Oracle Database-Specific Code versus Non-Oracle 3GL and 4GL code in Conjunction with SQL and PL/SQL: Key Differentiators  
15

1.3.1 Data Processing Differentiators  
16
1.3.2 Data Sharing Differentiators 17
1.3.3 Code Manageability Differentiators 18
1.3.4 Error Handling Differentiators 19
1.3.5 Other Key Differentiators 19

Chapter 2 Feature-Set and Solution-Set Enhancements 23

2.1 Introduction 23
2.2 New Feature and Solution Options in Oracle11g 23
   2.2.1 SQL/PL/SQL 23
   2.2.2 New Functionalities That Benefit the Enterprise Grid 31
   2.2.3 PHP/Ajax 32
   2.2.4 Oracle, .NET and C# 33
   2.2.5 Java 34
   2.2.6 XML 35
2.3 New Feature and Solution Options in Oracle10g 35
   2.3.1 Regular Expressions 35
   2.3.2 DBMS_JAVA.SET_OUTPUT 36
   2.3.3 Tracking the Error Line Number 36
   2.3.4 Enhanced FORALL Statement for Handling DML Array Processing with Sparse Collections 39
   2.3.5 Conditional Compilation (Oracle 10GR2) 40
   2.3.6 Compile-Time Warnings 41
   2.3.7 Integrated mod_plsql 42

Chapter 3 Programming Languages, Platforms, and Solutions 43

3.1 Introduction 43
3.2 Why and When Architecture Takes Precedence over Feature Set 44
   3.2.1 Suitability Criteria from a Business/Customer Perspective 49
   3.2.2 Suitability Criteria from a Solution Architecture (Technical/Techno-Functional) Perspective 50
3.3 Best Practices in Terms of Oracle Database Interaction 53
   3.3.1 Enforcing Business Rules for Efficiency 56
   3.3.2 Pre-emptive and Proactive Measures 57
3.3.3 Database Interaction: More than Just GB and Data Quality 58

3.4 Best Practices for using SQL and PL/SQL only (with HTML, XML via PSP or Web Toolkit) 59

3.5 Extending PL/SQL to Use Emerging Dynamic Scripting Languages Like Groovy in the Database 61
  3.5.1 Extending PL/SQL to Use HTML/XML 61
  3.5.2 Generating Web-Enabled PL/SQL Applications Using Oracle APEX/Oracle JDeveloper 65

3.6 Best Practices for Using .NET-Based Languages (C++, C#, J#) 66
  3.6.1 ODP.NET: More than Just ADO.NET Rehashed 66
  3.6.2 Leverage J2EE Inherent Server-Specific Benefits with Minimal Cost and Time-to-Deliver 67

3.7 Best Practices for Using Java-Based Languages (Java Libraries, Java via JDBC, Java via SQLJ, Java via Java Stored Procedures) 69
  3.7.1 Using Java via JDBC, SQLJ, Java Stored Procedures or Java Libraries 73

3.8 Best Practices for Using PHP-Based Languages (PHP-only, PHP and Ajax, PHP Web Pages) 75

3.9 Perl in the Picture: Best Practices for the Perfect Fit and Use of Perl in an Embedded Scenario 77

3.10 For Better or Worse: Application-centric and Business-centric Key Performance Indicators 78
  3.10.1 Focus on the Customer 78
  3.10.2 Application-Centric 79

Chapter 4 Best Practices for Data Structure Management 83

4.1 Introduction 83

4.2 Data Representation: Database versus Programming Language Perspectives 85
  4.2.1 Database Implementation 93
  4.2.2 Programming Language Implementation 94

4.3 Best Practices for Using Heterogeneous Data Structures: Arrays, Array Lists, Lists, Maps, Enumerations 95
6.2 Database Management Using Embedded Programming Languages: Design Practices for the Best Degree of Fit 140
6.3 Best Practices in Terms of Connectivity to and Interaction with the Oracle Database 144
6.4 Techniques to Connect to Multiple Databases Simultaneously Using the Same Metadata 148
6.5 Best Fit for Oracle in the ADO.NET Framework—ODP.NET and ODE.NET 150
6.6 Best Practices for Data Retrieval 150
6.7 Best Practices for Data Manipulation 153
6.8 Best Practices for Data Validation 155
6.9 Best Practices for using LINQ to SQL 158
6.10 Best Practices for Using XML 158
6.11 Best Practices for Handling Unstructured Data 162
6.12 Best Practices to Protect Data Integrity and Special Cases Involved 165
   6.12.1 User Interaction with Data (and hence with the Database) 166
   6.12.2 Special Cases Involved 171
   6.12.3 Application Interaction with Data (and hence with the Database) 171
   6.12.4 Design Pattern for a Typical Code Accelerator 176

Chapter 7 Best Practices for Application Management 181
7.1 Introduction 181
7.2 Code Accelerators for Rapid Application Development: Design and Coding Aspects 183
   7.2.1 Code Accelerator Design Pragmatics 187
7.3 Best Practices for Application Web Interaction: Receiving, Routing, Responding, Redirecting, Rendering, Linking 196
   7.3.1 Personalized A-La-Carte of Customer/End-User Service Requests: A Real-World Use Case of Application Web Interaction 199
7.4 Best Practices for Application Integrity 200
   7.4.1 A Design Pattern for a Common Information Integrity Framework at the End-to-End Solution Level 201
7.5 Best Practices for Application Consistency 204
  7.5.1 Multitenancy of Data Services Calls, Web Services Calls, and Java Services Calls 205
  7.5.2 A Real-World Scenario that Demonstrates the Use of These Best Practices for Implementation 208
  7.6.1 Real-World Use Cases for Implementing Web Security Policies 214
7.7 Best Fit for Oracle in the ASP.NET Framework 216
7.8 Best Practices for Templates: Use or Eliminate? 220
7.9 Best Practices Beyond HTML: Auto-generate Dynamic Visual Content, Web Services, and Mobile Applications 221
  7.9.1 Auto-generate Dynamic Visual Content 221
  7.9.2 Web Services 225
  7.9.3 Mobilizing Applications 226
7.10 Best Practices for Creative Reporting 227

Chapter 8 Application Development Frameworks 237
8.1 Introduction 237
8.2 Application Development Framework: A Pragmatic "Best-Possible" Solution 238
  8.2.1 For a Transactional Solution 240
  8.2.2 For a Reporting and Analysis Solution 243
8.3 Master Error Management Framework 251
  8.3.1 Error-Logging Framework 251
  8.3.2 DML Auditing Framework 261
8.4 Performance Tuning Framework 273
  8.4.1 Framework for Monitoring and Tuning SQL 274
  8.4.2 Framework for Monitoring and Tuning PL/SQL 277
8.5 Debugging Framework 280
  8.5.1 Putting a Robust Error-Handling Mechanism in Place 281
  8.5.2 Introducing Debug Messages as Part of Code Asynchronously 281
  8.5.3 Testing of the Application Solution 294
### Chapter 9  Miscellaneous Best Practices

9.1 **Simulating Oracle-Based Datasets: Best Practices for Design and Coding**  
9.1.1 **The In-Memory Processing Framework**  
9.1.2 **Simulating Oracle-Based Data Sets Using Transformation Pipeline Chain**  
9.2 **Building a Highly Visible Web Site**

### Chapter 10  Best Practices in Terms of Coding Standards and Troubleshooting

10.1 **Introduction**  
10.2 **Coding Standards from an Oracle11g Embedded Programming Perspective**  
10.3 **Tuning SQL and PL/SQL Code for Optimality**  
10.4 **Tuning Embedded Language Code for Optimality: Data Structures and Procedural Code**  
10.5 **Fine-Tuning Solution Methodology for Optimality**

### Index