

Oracle® Database

Database PL/SQL Language Reference



23ai
F46753-09
April 2025

ORACLE®

Copyright © 1996, 2025, Oracle and/or its affiliates.

Primary Author: Sarah Hirschfeld

Contributing Authors: P. Huey, L. Jayapalan

Contributors: D. Alpern, S. Agrawal, M. Bach, H. Baer, S. Castledine, T. Chang, B. Cheng, R. Dani, R. Decker, M. DiPaolo, C. Iyer, A. Kruglikov, N. Le, W. Li, P. Miller, V. Moore, T. Raney, R. Rajagopalan, C. Saxon, I. Stocks, C. Wetherell, S. Wolicki, G. Viswanathan, M. Yang

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Audience	xxxv
Documentation Accessibility	xxxv
Related Documents	xxxvi
Conventions	xxxvi
Syntax Descriptions	xxxvii

1 Changes in This Release for Oracle Database PL/SQL Language Reference

New Features in Release 23ai for Oracle Database PL/SQL Language Reference	1-1
SQL BOOLEAN Data Type	1-1
SQL VECTOR Data Type	1-3
IF [NOT] EXISTS Syntax Support	1-3
Extended CASE Controls	1-4
JSON Constructor and JSON_VALUE Support of PL/SQL Aggregate Types	1-5
SQL Transpiler	1-5
Oracle Database 23ai, Release Update 23.7	1-6
Jaccard Distance Metric	1-6
BINARY Vector Support	1-6
VECTOR Dimension-wise Arithmetic Support	1-6
Oracle Database 23ai, Release Update 23.8	1-7
SPARSE Vector Support	1-7
Deprecated Features	1-7
Desupported Features	1-7

2 Overview of PL/SQL

Advantages of PL/SQL	2-1
Tight Integration with SQL	2-1
High Performance	2-2
High Productivity	2-2
Portability	2-2
Scalability	2-3

Manageability	2-3
Support for Object-Oriented Programming	2-3
Main Features of PL/SQL	2-3
Error Handling	2-4
Blocks	2-4
Variables and Constants	2-5
Subprograms	2-5
Packages	2-5
Triggers	2-5
Input and Output	2-6
Data Abstraction	2-7
Cursors	2-7
Composite Variables	2-7
Using the %ROWTYPE Attribute	2-7
Using the %TYPE Attribute	2-8
Abstract Data Types	2-8
Control Statements	2-8
Conditional Compilation	2-9
Processing a Query Result Set One Row at a Time	2-9
Architecture of PL/SQL	2-10
PL/SQL Engine	2-10
PL/SQL Units and Compilation Parameters	2-11
Protecting Sensitive Information in PL/SQL	2-12

3 PL/SQL Language Fundamentals

Character Sets	3-1
Database Character Set	3-1
National Character Set	3-3
About Data-Bound Collation	3-3
Lexical Units	3-4
Delimiters	3-4
Identifiers	3-5
Reserved Words and Keywords	3-6
Predefined Identifiers	3-6
User-Defined Identifiers	3-6
Literals	3-10
Pragmas	3-12
Comments	3-12
Single-Line Comments	3-13
Multiline Comments	3-13
Whitespace Characters Between Lexical Units	3-14

Declarations	3-15
NOT NULL Constraint	3-15
Declaring Variables	3-16
Declaring Constants	3-16
Initial Values of Variables and Constants	3-17
Declaring Items using the %TYPE Attribute	3-18
References to Identifiers	3-19
Scope and Visibility of Identifiers	3-20
Assigning Values to Variables	3-24
Assigning Values to Variables with the Assignment Statement	3-25
Assigning Values to Variables with the SELECT INTO Statement	3-26
Assigning Values to Variables as Parameters of a Subprogram	3-26
Assigning Values to BOOLEAN Variables	3-27
Expressions	3-27
Concatenation Operator	3-28
Operator Precedence	3-29
Logical Operators	3-31
Short-Circuit Evaluation	3-36
Comparison Operators	3-36
IS [NOT] NULL Operator	3-37
Relational Operators	3-37
LIKE Operator	3-39
BETWEEN Operator	3-41
IN Operator	3-41
BOOLEAN Expressions	3-42
CASE Expressions	3-43
Simple CASE Expression	3-43
Searched CASE Expression	3-46
SQL Functions in PL/SQL Expressions	3-47
Static Expressions	3-48
PLS_INTEGER Static Expressions	3-51
BOOLEAN Static Expressions	3-51
VARCHAR2 Static Expressions	3-52
Static Constants	3-53
Error-Reporting Functions	3-54
Conditional Compilation	3-54
How Conditional Compilation Works	3-55
Preprocessor Control Tokens	3-55
Selection Directives	3-56
Error Directives	3-56
Inquiry Directives	3-57
DBMS_DB_VERSION Package	3-60

Conditional Compilation Examples	3-61
Retrieving and Printing Post-Processed Source Text	3-62
Conditional Compilation Directive Restrictions	3-63

4 PL/SQL Data Types

SQL Data Types	4-1
Different Maximum Sizes	4-2
Additional PL/SQL Constants for BINARY_FLOAT and BINARY_DOUBLE	4-2
Additional PL/SQL Subtypes of BINARY_FLOAT and BINARY_DOUBLE	4-3
BOOLEAN Data Type	4-3
JSON Data Type	4-5
PL/SQL and JSON Type Conversions	4-5
VECTOR Data Type	4-14
VECTOR Operations Supported by PL/SQL	4-17
VECTOR Data Type PL/SQL Code Examples	4-19
CHAR and VARCHAR2 Variables	4-23
Assigning or Inserting Too-Long Values	4-23
Declaring Variables for Multibyte Characters	4-24
Differences Between CHAR and VARCHAR2 Data Types	4-24
LONG and LONG RAW Variables	4-26
ROWID and UROWID Variables	4-26
PLS_INTEGER and BINARY_INTEGER Data Types	4-27
Preventing PLS_INTEGER Overflow	4-27
Predefined PLS_INTEGER Subtypes	4-28
SIMPLE_INTEGER Subtype of PLS_INTEGER	4-29
SIMPLE_INTEGER Overflow Semantics	4-30
Expressions with Both SIMPLE_INTEGER and Other Operands	4-30
Integer Literals in SIMPLE_INTEGER Range	4-30
User-Defined PL/SQL Subtypes	4-31
Unconstrained Subtypes	4-31
Constrained Subtypes	4-32
Subtypes with Base Types in Same Data Type Family	4-34

5 PL/SQL Control Statements

Conditional Selection Statements	5-1
IF THEN Statement	5-1
IF THEN ELSE Statement	5-3
IF THEN ELSIF Statement	5-4
Simple CASE Statement	5-6
Searched CASE Statement	5-8

LOOP Statements	5-9
Basic LOOP Statement	5-10
FOR LOOP Statement Overview	5-10
FOR LOOP Iterand	5-12
Iterand Mutability	5-14
Multiple Iteration Controls	5-14
Stepped Range Iteration Controls	5-15
Single Expression Iteration Controls	5-17
Collection Iteration Controls	5-18
Cursor Iteration Controls	5-21
Using Dynamic SQL in Iteration Controls	5-21
Stopping and Skipping Predicate Clauses	5-22
WHILE LOOP Statement	5-23
Sequential Control Statements	5-24
GOTO Statement	5-24
NULL Statement	5-24

6 PL/SQL Collections and Records

Collection Types	6-2
Associative Arrays	6-4
Declaring Associative Array Constants	6-7
NLS Parameter Values Affect Associative Arrays Indexed by String	6-8
Changing NLS Parameter Values After Populating Associative Arrays	6-9
Indexes of Data Types Other Than VARCHAR2	6-9
Passing Associative Arrays to Remote Databases	6-9
Appropriate Uses for Associative Arrays	6-10
Varrays (Variable-Size Arrays)	6-10
Appropriate Uses for Varrays	6-12
Nested Tables	6-13
Important Differences Between Nested Tables and Arrays	6-16
Appropriate Uses for Nested Tables	6-16
Collection Constructors	6-16
Qualified Expressions Overview	6-18
Assigning Values to Collection Variables	6-23
Data Type Compatibility	6-24
Assigning Null Values to Varray or Nested Table Variables	6-24
Assigning Set Operation Results to Nested Table Variables	6-25
Multidimensional Collections	6-27
Collection Comparisons	6-29
Comparing Varray and Nested Table Variables to NULL	6-29
Comparing Nested Tables for Equality and Inequality	6-30

Comparing Nested Tables with SQL Multiset Conditions	6-31
Collection Methods	6-32
DELETE Collection Method	6-33
TRIM Collection Method	6-36
EXTEND Collection Method	6-38
EXISTS Collection Method	6-39
FIRST and LAST Collection Methods	6-39
FIRST and LAST Methods for Associative Array	6-40
FIRST and LAST Methods for Varray	6-41
FIRST and LAST Methods for Nested Table	6-42
COUNT Collection Method	6-44
COUNT Method for Varray	6-44
COUNT Method for Nested Table	6-44
LIMIT Collection Method	6-45
PRIOR and NEXT Collection Methods	6-46
Collection Types Defined in Package Specifications	6-49
Record Variables	6-50
Initial Values of Record Variables	6-51
Declaring Record Constants	6-51
RECORD Types	6-52
Declaring Items using the %ROWTYPE Attribute	6-56
Declaring a Record Variable that Always Represents Full Row	6-56
Declaring a Record Variable that Can Represent Partial Row	6-58
%ROWTYPE Attribute and Virtual Columns	6-59
%ROWTYPE Attribute and Invisible Columns	6-60
Assigning Values to Record Variables	6-62
Assigning Values to RECORD Type Variables Using Qualified Expressions	6-62
Assigning One Record Variable to Another	6-63
Assigning Full or Partial Rows to Record Variables	6-65
Using SELECT INTO to Assign a Row to a Record Variable	6-66
Using FETCH to Assign a Row to a Record Variable	6-66
Using SQL Statements to Return Rows in PL/SQL Record Variables	6-68
Assigning NULL to a Record Variable	6-68
Record Comparisons	6-69
Inserting Records into Tables	6-69
Updating Rows with Records	6-70
Restrictions on Record Inserts and Updates	6-71

7 PL/SQL Static SQL

Description of Static SQL	7-1
Statements	7-1

Pseudocolumns	7-3
CURRVAL and NEXTVAL in PL/SQL	7-3
Cursors Overview	7-5
Implicit Cursors	7-6
SQL%ISOPEN Attribute: Is the Cursor Open?	7-7
SQL%FOUND Attribute: Were Any Rows Affected?	7-7
SQL%NOTFOUND Attribute: Were No Rows Affected?	7-8
SQL%ROWCOUNT Attribute: How Many Rows Were Affected?	7-8
Explicit Cursors	7-9
Declaring and Defining Explicit Cursors	7-9
Opening and Closing Explicit Cursors	7-10
Fetching Data with Explicit Cursors	7-11
Variables in Explicit Cursor Queries	7-13
When Explicit Cursor Queries Need Column Aliases	7-15
Explicit Cursors that Accept Parameters	7-16
Explicit Cursor Attributes	7-20
Processing Query Result Sets	7-24
Processing Query Result Sets With SELECT INTO Statements	7-25
Handling Single-Row Result Sets	7-25
Handling Large Multiple-Row Result Sets	7-25
Processing Query Result Sets With Cursor FOR LOOP Statements	7-25
Processing Query Result Sets With Explicit Cursors, OPEN, FETCH, and CLOSE	7-28
Processing Query Result Sets with Subqueries	7-28
Cursor Variables	7-30
Creating Cursor Variables	7-31
Opening and Closing Cursor Variables	7-32
Fetching Data with Cursor Variables	7-33
Assigning Values to Cursor Variables	7-35
Variables in Cursor Variable Queries	7-35
Querying a Collection	7-37
Cursor Variable Attributes	7-38
Cursor Variables as Subprogram Parameters	7-38
Cursor Variables as Host Variables	7-41
CURSOR Expressions	7-42
Transaction Processing and Control	7-43
COMMIT Statement	7-44
ROLLBACK Statement	7-46
SAVEPOINT Statement	7-47
Implicit Rollbacks	7-49
SET TRANSACTION Statement	7-49
Overriding Default Locking	7-50
LOCK TABLE Statement	7-51

SELECT FOR UPDATE and FOR UPDATE Cursors	7-51
Simulating CURRENT OF Clause with ROWID Pseudocolumn	7-52
Autonomous Transactions	7-54
Advantages of Autonomous Transactions	7-55
Transaction Context	7-55
Transaction Visibility	7-55
Declaring Autonomous Routines	7-56
Controlling Autonomous Transactions	7-57
Entering and Exiting Autonomous Routines	7-57
Committing and Rolling Back Autonomous Transactions	7-58
Savepoints	7-58
Avoiding Errors with Autonomous Transactions	7-58
Autonomous Triggers	7-58
Invoking Autonomous Functions from SQL	7-61

8 PL/SQL Dynamic SQL

When You Need Dynamic SQL	8-1
Native Dynamic SQL	8-2
EXECUTE IMMEDIATE Statement	8-2
OPEN FOR, FETCH, and CLOSE Statements	8-8
Repeated Placeholder Names in Dynamic SQL Statements	8-10
Dynamic SQL Statement is Not Anonymous Block or CALL Statement	8-10
Dynamic SQL Statement is Anonymous Block or CALL Statement	8-10
DBMS_SQL Package	8-11
DBMS_SQL.RETURN_RESULT Procedure	8-12
DBMS_SQL.GET_NEXT_RESULT Procedure	8-14
DBMS_SQL.TO_REFCURSOR Function	8-15
DBMS_SQL.TO_CURSOR_NUMBER Function	8-17
SQL Injection	8-18
SQL Injection Techniques	8-18
Statement Modification	8-19
Statement Injection	8-20
Data Type Conversion	8-22
Guards Against SQL Injection	8-23
Bind Variables	8-24
Validation Checks	8-25
Explicit Format Models	8-27

9 PL/SQL Subprograms

Reasons to Use Subprograms	9-1
----------------------------	-----

Nested, Package, and Standalone Subprograms	9-2
Subprogram Invocations	9-2
Subprogram Properties	9-3
Subprogram Parts	9-3
Additional Parts for Functions	9-5
RETURN Statement	9-6
RETURN Statement in Function	9-6
RETURN Statement in Procedure	9-8
RETURN Statement in Anonymous Block	9-8
Forward Declaration	9-9
Subprogram Parameters	9-9
Formal and Actual Subprogram Parameters	9-9
Formal Parameters of Constrained Subtypes	9-11
Subprogram Parameter Passing Methods	9-13
Subprogram Parameter Modes	9-14
Subprogram Parameter Aliasing	9-20
Subprogram Parameter Aliasing with Parameters Passed by Reference	9-20
Subprogram Parameter Aliasing with Cursor Variable Parameters	9-22
Default Values for IN Subprogram Parameters	9-23
Positional, Named, and Mixed Notation for Actual Parameters	9-25
Subprogram Invocation Resolution	9-27
Overloaded Subprograms	9-29
Formal Parameters that Differ Only in Numeric Data Type	9-30
Subprograms that You Cannot Overload	9-31
Subprogram Overload Errors	9-32
Recursive Subprograms	9-35
Subprogram Side Effects	9-37
PL/SQL Function Result Cache	9-37
Enabling Result-Caching for a Function	9-38
Developing Applications with Result-Cached Functions	9-40
Requirements for Result-Cached Functions	9-40
Examples of Result-Cached Functions	9-41
Result-Cached Application Configuration Parameters	9-41
Result-Cached Recursive Function	9-43
Advanced Result-Cached Function Topics	9-43
Rules for a Cache Hit	9-43
Result Cache Bypass	9-44
Making Result-Cached Functions Handle Session-Specific Settings	9-44
Making Result-Cached Functions Handle Session-Specific Application Contexts	9-45
Choosing Result-Caching Granularity	9-46
Result Caches in Oracle RAC Environment	9-48
Result Cache Management	9-49

Hot-Patching PL/SQL Units on Which Result-Cached Functions Depend	9-50
PL/SQL Functions that SQL Statements Can Invoke	9-51
Invoker's Rights and Definer's Rights (AUTHID Property)	9-52
Granting Roles to PL/SQL Packages and Standalone Subprograms	9-54
IR Units Need Template Objects	9-54
Connected User Database Links in DR Units	9-54
External Subprograms	9-55

10 PL/SQL Triggers

Overview of Triggers	10-1
Reasons to Use Triggers	10-2
DML Triggers	10-4
Conditional Predicates for Detecting Triggering DML Statement	10-5
INSTEAD OF DML Triggers	10-5
Compound DML Triggers	10-10
Compound DML Trigger Structure	10-10
Compound DML Trigger Restrictions	10-11
Performance Benefit of Compound DML Triggers	10-11
Using Compound DML Triggers with Bulk Insertion	10-12
Using Compound DML Triggers to Avoid Mutating-Table Error	10-15
Triggers for Ensuring Referential Integrity	10-15
Foreign Key Trigger for Child Table	10-17
UPDATE and DELETE RESTRICT Trigger for Parent Table	10-18
UPDATE and DELETE SET NULL Trigger for Parent Table	10-19
DELETE CASCADE Trigger for Parent Table	10-19
UPDATE CASCADE Trigger for Parent Table	10-20
Triggers for Complex Constraint Checking	10-21
Triggers for Complex Security Authorizations	10-22
Triggers for Transparent Event Logging	10-24
Triggers for Deriving Column Values	10-24
Triggers for Building Complex Updatable Views	10-24
Triggers for Fine-Grained Access Control	10-27
Correlation Names and Pseudorecords	10-28
OBJECT_VALUE Pseudocolumn	10-33
System Triggers	10-34
SCHEMA Triggers	10-35
DATABASE Triggers	10-35
INSTEAD OF CREATE Triggers	10-36
Subprograms Invoked by Triggers	10-36
Trigger Compilation, Invalidation, and Recompile	10-37
Exception Handling in Triggers	10-38

Trigger Design Guidelines	10-39
Trigger Restrictions	10-41
Trigger Size Restriction	10-41
Trigger LONG and LONG RAW Data Type Restrictions	10-42
Mutating-Table Restriction	10-42
Order in Which Triggers Fire	10-46
Trigger Enabling and Disabling	10-47
Trigger Changing and Debugging	10-47
Triggers and Oracle Database Data Transfer Utilities	10-48
Triggers for Publishing Events	10-49
Event Attribute Functions	10-50
Event Attribute Functions for Database Event Triggers	10-54
Event Attribute Functions for Client Event Triggers	10-55
Views for Information About Triggers	10-60

11 PL/SQL Packages

What is a Package?	11-1
Reasons to Use Packages	11-2
Package Specification	11-3
Appropriate Public Items	11-3
Creating Package Specifications	11-4
Package Body	11-5
Package Instantiation and Initialization	11-7
Package State	11-7
SERIALLY_REUSABLE Packages	11-8
Creating SERIALLY_REUSABLE Packages	11-9
SERIALLY_REUSABLE Package Work Unit	11-9
Explicit Cursors in SERIALLY_REUSABLE Packages	11-10
Package Writing Guidelines	11-12
Package Example	11-14
How STANDARD Package Defines the PL/SQL Environment	11-18

12 PL/SQL Error Handling

Compile-Time Warnings	12-2
DBMS_WARNING Package	12-3
Overview of Exception Handling	12-5
Exception Categories	12-6
Advantages of Exception Handlers	12-7
Guidelines for Avoiding and Handling Exceptions	12-9
Internally Defined Exceptions	12-9

Predefined Exceptions	12-11
User-Defined Exceptions	12-13
Redeclared Predefined Exceptions	12-13
Raising Exceptions Explicitly	12-15
RAISE Statement	12-15
Raising User-Defined Exception with RAISE Statement	12-15
Raising Internally Defined Exception with RAISE Statement	12-16
Reraising Current Exception with RAISE Statement	12-17
RAISE_APPLICATION_ERROR Procedure	12-18
Exception Propagation	12-19
Propagation of Exceptions Raised in Declarations	12-22
Propagation of Exceptions Raised in Exception Handlers	12-23
Unhandled Exceptions	12-27
Retrieving Error Code and Error Message	12-27
Continuing Execution After Handling Exceptions	12-28
Retrying Transactions After Handling Exceptions	12-30
Handling Errors in Distributed Queries	12-31

13 PL/SQL Optimization and Tuning

PL/SQL Optimizer	13-1
Subprogram Inlining	13-2
Candidates for Tuning	13-4
Minimizing CPU Overhead	13-5
Tune SQL Statements	13-5
Tune Function Invocations in Queries	13-5
Tune Subprogram Invocations	13-7
Tune Loops	13-8
Tune Computation-Intensive PL/SQL Code	13-9
Use Data Types that Use Hardware Arithmetic	13-9
Avoid Constrained Subtypes in Performance-Critical Code	13-10
Minimize Implicit Data Type Conversion	13-10
Use SQL Character Functions	13-11
Put Least Expensive Conditional Tests First	13-11
Bulk SQL and Bulk Binding	13-11
FORALL Statement	13-12
Using FORALL Statements for Sparse Collections	13-15
Unhandled Exceptions in FORALL Statements	13-18
Handling FORALL Exceptions Immediately	13-19
Handling FORALL Exceptions After FORALL Statement Completes	13-20
Getting Number of Rows Affected by FORALL Statement	13-23
BULK COLLECT Clause	13-25

SELECT INTO Statement with BULK COLLECT Clause	13-26
FETCH Statement with BULK COLLECT Clause	13-33
RETURNING INTO Clause with BULK COLLECT Clause	13-38
Using FORALL Statement and BULK COLLECT Clause Together	13-39
Client Bulk-Binding of Host Arrays	13-41
Chaining Pipelined Table Functions for Multiple Transformations	13-42
Overview of Table Functions	13-42
Creating Pipelined Table Functions	13-43
Pipelined Table Functions as Transformation Functions	13-46
Chaining Pipelined Table Functions	13-47
Fetching from Results of Pipelined Table Functions	13-47
Passing CURSOR Expressions to Pipelined Table Functions	13-48
DML Statements on Pipelined Table Function Results	13-51
NO_DATA_NEEDED Exception	13-51
Overview of Polymorphic Table Functions	13-53
Polymorphic Table Function Definition	13-54
Polymorphic Table Function Implementation	13-54
Polymorphic Table Function Invocation	13-55
Variadic Pseudo-Operators	13-56
COLUMNS Pseudo-Operator	13-57
Polymorphic Table Function Compilation and Execution	13-57
Polymorphic Table Function Optimization	13-57
Skip_col Polymorphic Table Function Example	13-58
To_doc Polymorphic Table Function Example	13-61
Implicit_echo Polymorphic Table Function Example	13-65
Updating Large Tables in Parallel	13-67
Collecting Data About User-Defined Identifiers	13-68
Profiling and Tracing PL/SQL Programs	13-68
Compiling PL/SQL Units for Native Execution	13-70
Determining Whether to Use PL/SQL Native Compilation	13-70
How PL/SQL Native Compilation Works	13-71
Dependencies, Invalidation, and Revalidation	13-71
Setting Up a New Database for PL/SQL Native Compilation	13-71
Compiling the Entire Database for PL/SQL Native or Interpreted Compilation	13-72

14 PL/SQL Language Elements

ACCESSIBLE BY Clause	14-3
AGGREGATE Clause	14-8
Assignment Statement	14-9
AUTONOMOUS_TRANSACTION Pragma	14-12
Basic LOOP Statement	14-13

Block	14-15
Call Specification	14-24
CASE Statement	14-27
CLOSE Statement	14-30
Collection Method Invocation	14-31
Collection Variable Declaration	14-34
Comment	14-39
COMPILE Clause	14-40
Constant Declaration	14-43
CONTINUE Statement	14-44
COVERAGE Pragma	14-46
Cursor FOR LOOP Statement	14-49
Cursor Variable Declaration	14-51
Datatype Attribute	14-54
DEFAULT COLLATION Clause	14-55
DELETE Statement Extension	14-57
DEPRECATE Pragma	14-57
DETERMINISTIC Clause	14-67
Element Specification	14-68
EXCEPTION_INIT Pragma	14-74
Exception Declaration	14-76
Exception Handler	14-77
EXECUTE IMMEDIATE Statement	14-79
EXIT Statement	14-82
Explicit Cursor Declaration and Definition	14-84
Expression	14-88
FETCH Statement	14-98
FOR LOOP Statement	14-100
FORALL Statement	14-103
Formal Parameter Declaration	14-105
Function Declaration and Definition	14-108
GOTO Statement	14-110
IF Statement	14-113
Implicit Cursor Attribute	14-115
INLINE Pragma	14-117
Invoker's Rights and Definer's Rights Clause	14-118
INSERT Statement Extension	14-119
Iterator	14-121
Named Cursor Attribute	14-128
NULL Statement	14-130
OPEN Statement	14-130
OPEN FOR Statement	14-132

PARALLEL_ENABLE Clause	14-134
PIPE ROW Statement	14-137
PIPELINED Clause	14-138
Procedure Declaration and Definition	14-141
Qualified Expression	14-143
RAISE Statement	14-148
Record Variable Declaration	14-149
RESTRICT_REFERENCES Pragma	14-151
RETURN Statement	14-153
RETURNING INTO Clause	14-154
RESULT_CACHE Clause	14-157
%ROWTYPE Attribute	14-159
Scalar Variable Declaration	14-161
SELECT INTO Statement	14-162
SERIALLY_REUSABLE Pragma	14-164
SHARD_ENABLE Clause	14-165
SHARING Clause	14-167
SQL_MACRO Clause	14-169
SQLCODE Function	14-177
SQLERRM Function	14-178
SUPPRESSES_WARNING_6009 Pragma	14-180
%TYPE Attribute	14-186
UDF Pragma	14-188
UPDATE Statement Extensions	14-188
WHILE LOOP Statement	14-189

15 SQL Statements for Stored PL/SQL Units

ALTER FUNCTION Statement	15-2
ALTER LIBRARY Statement	15-4
ALTER PACKAGE Statement	15-6
ALTER PROCEDURE Statement	15-8
ALTER TRIGGER Statement	15-10
ALTER TYPE Statement	15-13
CREATE FUNCTION Statement	15-24
CREATE LIBRARY Statement	15-31
CREATE PACKAGE Statement	15-35
CREATE PACKAGE BODY Statement	15-39
CREATE PROCEDURE Statement	15-43
CREATE TRIGGER Statement	15-48
CREATE TYPE Statement	15-67
CREATE TYPE BODY Statement	15-80

DROP FUNCTION Statement	15-84
DROP LIBRARY Statement	15-86
DROP PACKAGE Statement	15-87
DROP PROCEDURE Statement	15-89
DROP TRIGGER Statement	15-90
DROP TYPE Statement	15-91
DROP TYPE BODY Statement	15-93

A PL/SQL Source Text Wrapping

PL/SQL Source Text Wrapping Limitations	A-2
PL/SQL Source Text Wrapping Guidelines	A-2
Wrapping PL/SQL Source Text with PL/SQL Wrapper Utility	A-2
Wrapping PL/SQL Source Text with DBMS_DDL Subprograms	A-8

B PL/SQL Name Resolution

Qualified Names and Dot Notation	B-1
Column Name Precedence	B-3
Differences Between PL/SQL and SQL Name Resolution Rules	B-5
Resolution of Names in Static SQL Statements	B-6
What is Capture?	B-7
Outer Capture	B-7
Same-Scope Capture	B-7
Inner Capture	B-7
Avoiding Inner Capture in SELECT and DML Statements	B-8
Qualifying References to Attributes and Methods	B-9
Qualifying References to Row Expressions	B-10

C PL/SQL Program Limits

D PL/SQL Reserved Words and Keywords

E PL/SQL Predefined Data Types

Index

List of Examples

1-1	Calling a PL/SQL Function with BOOLEAN Argument from SQL	1-2
1-2	CREATE PROCEDURE with IF NOT EXISTS	1-4
2-1	PL/SQL Block Structure	2-4
2-2	Processing Query Result Rows One at a Time	2-9
3-1	Valid Case-Insensitive Reference to Quoted User-Defined Identifier	3-8
3-2	Invalid Case-Insensitive Reference to Quoted User-Defined Identifier	3-8
3-3	Reserved Word as Quoted User-Defined Identifier	3-9
3-4	Neglecting Double Quotation Marks	3-9
3-5	Neglecting Case-Sensitivity	3-10
3-6	Single-Line Comments	3-13
3-7	Multiline Comments	3-14
3-8	Whitespace Characters Improving Source Text Readability	3-14
3-9	Variable Declaration with NOT NULL Constraint	3-15
3-10	Variables Initialized to NULL Values	3-16
3-11	Scalar Variable Declarations	3-16
3-12	Constant Declarations	3-17
3-13	Variable and Constant Declarations with Initial Values	3-17
3-14	Variable Initialized to NULL by Default	3-17
3-15	Declaring Variable of Same Type as Column	3-19
3-16	Declaring Variable of Same Type as Another Variable	3-19
3-17	Scope and Visibility of Identifiers	3-20
3-18	Qualifying Redeclared Global Identifier with Block Label	3-21
3-19	Qualifying Identifier with Subprogram Name	3-22
3-20	Duplicate Identifiers in Same Scope	3-22
3-21	Declaring Same Identifier in Different Units	3-23
3-22	Label and Subprogram with Same Name in Same Scope	3-23
3-23	Block with Multiple and Duplicate Labels	3-24
3-24	Assigning Values to Variables with Assignment Statement	3-25
3-25	Assigning Value to Variable with SELECT INTO Statement	3-26
3-26	Assigning Value to Variable as IN OUT Subprogram Parameter	3-26
3-27	Assigning Value to BOOLEAN Variable	3-27
3-28	Concatenation Operator	3-28
3-29	Concatenation Operator with NULL Operands	3-28
3-30	Controlling Evaluation Order with Parentheses	3-29
3-31	Expression with Nested Parentheses	3-29
3-32	Improving Readability with Parentheses	3-30

3-33	Operator Precedence	3-30
3-34	Procedure Prints BOOLEAN Variable	3-31
3-35	AND Operator	3-32
3-36	OR Operator	3-33
3-37	NOT Operator	3-33
3-38	NULL Value in Unequal Comparison	3-34
3-39	NULL Value in Equal Comparison	3-34
3-40	NOT NULL Equals NULL	3-35
3-41	Changing Evaluation Order of Logical Operators	3-35
3-42	Short-Circuit Evaluation	3-36
3-43	Relational Operators in Expressions	3-38
3-44	LIKE Operator in Expression	3-40
3-45	Escape Character in Pattern	3-40
3-46	BETWEEN Operator in Expressions	3-41
3-47	IN Operator in Expressions	3-41
3-48	IN Operator with Sets with NULL Values	3-42
3-49	Equivalent BOOLEAN Expressions	3-43
3-50	Simple CASE Expression	3-44
3-51	Simple CASE Expression with WHEN NULL	3-44
3-52	Simple CASE Expression with List of selector_values	3-45
3-53	Simple CASE Expression with Dangling Predicates	3-45
3-54	Searched CASE Expression	3-46
3-55	Searched CASE Expression with WHEN ... IS NULL	3-47
3-56	Static Constants	3-54
3-57	Predefined Inquiry Directives	3-58
3-58	Displaying Values of PL/SQL Compilation Parameters	3-58
3-59	PLSQL_CCFLAGS Assigns Value to Itself	3-60
3-60	Code for Checking Database Version	3-61
3-61	Compiling Different Code for Different Database Versions	3-61
3-62	Displaying Post-Processed Source Textsource text	3-63
3-63	Using Conditional Compilation Directive in the Definition of a Package Specification	3-64
3-64	Using Conditional Compilation Directive in the Formal Parameter List of a Subprogram	3-64
4-1	Printing BOOLEAN Values	4-4
4-2	Convert a JSON Object to PL/SQL Records	4-7
4-3	Convert a PL/SQL Record to a JSON Object	4-8
4-4	Convert a JSON Object to an Index by PLS_INTEGER Collection	4-9
4-5	Convert a JSON Object to a Nested Table Collection	4-9

4-6	Convert an Index by PLS_INTEGER Collection to a JSON Object	4-10
4-7	Convert a Nested Table to a JSON Object	4-10
4-8	Convert a JSON Array to an Index by PLS_INTEGER Collection	4-11
4-9	Convert a JSON Array to a Varray	4-11
4-10	Convert a JSON Array to a Nested Table	4-12
4-11	Convert a Varray to a JSON Array	4-12
4-12	Convert a JSON Object to an Associative Array	4-13
4-13	Convert an Associative Array to a JSON Object	4-14
4-14	Use the VECTOR Data Type with PL/SQL	4-19
4-15	Use the VECTOR Data Type with a PL/SQL Trigger	4-20
4-16	Use Vector Distance Functions with PL/SQL	4-20
4-17	Use Shorthand Distance Operators	4-21
4-18	Perform Arithmetic Operations on Vectors	4-22
4-19	Declare DENSE and SPARSE Vectors in PL/SQL	4-22
4-20	CHAR and VARCHAR2 Blank-Padding Difference	4-25
4-21	PLS_INTEGER Calculation Raises Overflow Exception	4-28
4-22	Preventing Overflow	4-28
4-23	Violating Constraint of SIMPLE_INTEGER Subtype	4-29
4-24	User-Defined Unconstrained Subtypes Show Intended Use	4-31
4-25	User-Defined Constrained Subtype Detects Out-of-Range Values	4-33
4-26	Implicit Conversion Between Constrained Subtypes with Same Base Type	4-33
4-27	Implicit Conversion Between Subtypes with Base Types in Same Family	4-34
5-1	IF THEN Statement	5-2
5-2	IF THEN ELSE Statement	5-3
5-3	Nested IF THEN ELSE Statements	5-4
5-4	IF THEN ELSIF Statement	5-5
5-5	IF THEN ELSIF Statement Simulates Simple CASE Statement	5-6
5-6	Simple CASE Statement	5-7
5-7	Simple CASE Statement with Dangling Predicates	5-7
5-8	Searched CASE Statement	5-8
5-9	EXCEPTION Instead of ELSE Clause in CASE Statement	5-9
5-10	FOR LOOP Statement Tries to Change Index Value	5-12
5-11	Outside Statement References FOR LOOP Statement Index	5-12
5-12	FOR LOOP Statement Index with Same Name as Variable	5-13
5-13	FOR LOOP Statement References Variable with Same Name as Index	5-13
5-14	Nested FOR LOOP Statements with Same Index Name	5-13
5-15	Using Multiple Iteration Controls	5-15

5-16	FOR LOOP Statements Range Iteration Control	5-16
5-17	Reverse FOR LOOP Statements Range Iteration Control	5-16
5-18	Stepped Range Iteration Controls	5-16
5-19	STEP Clause in FOR LOOP Statement	5-16
5-20	Simple Step Filter Using FOR LOOP Stepped Range Iterator	5-17
5-21	Single Expression Iteration Control	5-17
5-22	VALUES OF Iteration Control	5-19
5-23	INDICES OF Iteration Control	5-20
5-24	PAIRS OF Iteration Control	5-20
5-25	Cursor Iteration Controls	5-21
5-26	Using Dynamic SQL As An Iteration Control	5-22
5-27	Using Dynamic SQL As An Iteration Control In a Qualified Expression	5-22
5-28	Using FOR LOOP Stopping Predicate Clause	5-22
5-29	Using FOR LOOP Skipping Predicate Clause	5-23
5-30	NULL Statement Showing No Action	5-25
5-31	NULL Statement as Placeholder During Subprogram Creation	5-25
5-32	NULL Statement in ELSE Clause of Simple CASE Statement	5-25
6-1	Associative Array Indexed by String	6-5
6-2	Function Returns Associative Array Indexed by PLS_INTEGER	6-6
6-3	Declaring Associative Array Constant	6-7
6-4	Varray (Variable-Size Array)	6-11
6-5	Nested Table of Local Type	6-13
6-6	Nested Table of Standalone Type	6-15
6-7	Initializing Collection (Varray) Variable to Empty	6-17
6-8	Iterator Choice Association in Qualified Expressions	6-20
6-9	Index Iterator Choice Association in Qualified Expressions	6-21
6-10	Sequence Iterator Choice Association in Qualified Expressions	6-21
6-11	Assigning Values to Associative Array Type Variables Using Qualified Expressions	6-21
6-12	Assigning values to a RECORD Type Variables using Qualified Expressions	6-22
6-13	Assigning Values to a VARRAY Type using Qualified Expressions	6-23
6-14	Data Type Compatibility for Collection Assignment	6-24
6-15	Assigning Null Value to Nested Table Variable	6-25
6-16	Assigning Set Operation Results to Nested Table Variable	6-25
6-17	Two-Dimensional Varray (Varray of Varrays)	6-27
6-18	Nested Tables of Nested Tables and Varrays of Integers	6-27
6-19	Nested Tables of Associative Arrays and Varrays of Strings	6-28
6-20	Comparing Varray and Nested Table Variables to NULL	6-29

6-21	Comparing Nested Tables for Equality and Inequality	6-30
6-22	Comparing Nested Tables with SQL Multiset Conditions	6-31
6-23	DELETE Method with Nested Table	6-34
6-24	DELETE Method with Associative Array Indexed by String	6-35
6-25	TRIM Method with Nested Table	6-37
6-26	EXTEND Method with Nested Table	6-38
6-27	EXISTS Method with Nested Table	6-39
6-28	FIRST and LAST Values for Associative Array Indexed by PLS_INTEGER	6-40
6-29	FIRST and LAST Values for Associative Array Indexed by String	6-41
6-30	Printing Varray with FIRST and LAST in FOR LOOP	6-41
6-31	Printing Nested Table with FIRST and LAST in FOR LOOP	6-42
6-32	COUNT and LAST Values for Varray	6-44
6-33	COUNT and LAST Values for Nested Table	6-44
6-34	LIMIT and COUNT Values for Different Collection Types	6-45
6-35	PRIOR and NEXT Methods	6-47
6-36	Printing Elements of Sparse Nested Table	6-48
6-37	Identically Defined Package and Local Collection Types	6-49
6-38	Identically Defined Package and Standalone Collection Types	6-50
6-39	Declaring Record Constant	6-51
6-40	Declaring Record Constant	6-52
6-41	RECORD Type Definition and Variable Declaration	6-53
6-42	RECORD Type with RECORD Field (Nested Record)	6-53
6-43	RECORD Type with Varray Field	6-54
6-44	Identically Defined Package and Local RECORD Types	6-55
6-45	%ROWTYPE Variable Represents Full Database Table Row	6-56
6-46	%ROWTYPE Variable Does Not Inherit Initial Values or Constraints	6-57
6-47	%ROWTYPE Variable Represents Partial Database Table Row	6-58
6-48	%ROWTYPE Variable Represents Join Row	6-59
6-49	Inserting %ROWTYPE Record into Table (Wrong)	6-59
6-50	Inserting %ROWTYPE Record into Table (Right)	6-60
6-51	%ROWTYPE Affected by Making Invisible Column Visible	6-61
6-52	Assigning Values to RECORD Type Variables Using Qualified Expressions	6-62
6-53	Assigning Record to Another Record of Same RECORD Type	6-64
6-54	Assigning %ROWTYPE Record to RECORD Type Record	6-64
6-55	Assigning Nested Record to Another Record of Same RECORD Type	6-65
6-56	SELECT INTO Assigns Values to Record Variable	6-66
6-57	FETCH Assigns Values to Record that Function Returns	6-67

6-58	UPDATE Statement Assigns Values to Record Variable	6-68
6-59	Assigning NULL to Record Variable	6-68
6-60	Initializing Table by Inserting Record of Default Values	6-69
6-61	Updating Rows with Record	6-71
7-1	Static SQL Statements	7-2
7-2	CURRVAL and NEXTVAL Pseudocolumns	7-4
7-3	SQL%FOUND Implicit Cursor Attribute	7-7
7-4	SQL%ROWCOUNT Implicit Cursor Attribute	7-8
7-5	Explicit Cursor Declaration and Definition	7-10
7-6	FETCH Statements Inside LOOP Statements	7-11
7-7	Fetching Same Explicit Cursor into Different Variables	7-12
7-8	Variable in Explicit Cursor Query—No Result Set Change	7-13
7-9	Variable in Explicit Cursor Query—Result Set Change	7-14
7-10	Explicit Cursor with Virtual Column that Needs Alias	7-15
7-11	Explicit Cursor that Accepts Parameters	7-16
7-12	Cursor Parameters with Default Values	7-17
7-13	Adding Formal Parameter to Existing Cursor	7-19
7-14	%ISOPEN Explicit Cursor Attribute	7-21
7-15	%FOUND Explicit Cursor Attribute	7-21
7-16	%NOTFOUND Explicit Cursor Attribute	7-22
7-17	%ROWCOUNT Explicit Cursor Attribute	7-23
7-18	Implicit Cursor FOR LOOP Statement	7-26
7-19	Explicit Cursor FOR LOOP Statement	7-27
7-20	Passing Parameters to Explicit Cursor FOR LOOP Statement	7-27
7-21	Cursor FOR Loop References Virtual Columns	7-28
7-22	Subquery in FROM Clause of Parent Query	7-28
7-23	Correlated Subquery	7-29
7-24	Cursor Variable Declarations	7-31
7-25	Cursor Variable with User-Defined Return Type	7-32
7-26	Fetching Data with Cursor Variables	7-33
7-27	Fetching from Cursor Variable into Collections	7-34
7-28	Variable in Cursor Variable Query—No Result Set Change	7-35
7-29	Variable in Cursor Variable Query—Result Set Change	7-36
7-30	Querying a Collection with Static SQL	7-38
7-31	Procedure to Open Cursor Variable for One Query	7-39
7-32	Opening Cursor Variable for Chosen Query (Same Return Type)	7-40
7-33	Opening Cursor Variable for Chosen Query (Different Return Types)	7-40

7-34	Cursor Variable as Host Variable in Pro*C Client Program	7-42
7-35	CURSOR Expression	7-42
7-36	COMMIT Statement with COMMENT and WRITE Clauses	7-45
7-37	ROLLBACK Statement	7-46
7-38	SAVEPOINT and ROLLBACK Statements	7-48
7-39	Reusing SAVEPOINT with ROLLBACK	7-48
7-40	SET TRANSACTION Statement in Read-Only Transaction	7-50
7-41	FETCH with FOR UPDATE Cursor After COMMIT Statement	7-52
7-42	Simulating CURRENT OF Clause with ROWID Pseudocolumn	7-53
7-43	Declaring Autonomous Function in Package	7-56
7-44	Declaring Autonomous Standalone Procedure	7-56
7-45	Declaring Autonomous PL/SQL Block	7-57
7-46	Autonomous Trigger Logs INSERT Statements	7-59
7-47	Autonomous Trigger Uses Native Dynamic SQL for DDL	7-60
7-48	Invoking Autonomous Function	7-61
8-1	Invoking Subprogram from Dynamic PL/SQL Block	8-4
8-2	Dynamically Invoking Subprogram with BOOLEAN Formal Parameter	8-4
8-3	Dynamically Invoking Subprogram with RECORD Formal Parameter	8-5
8-4	Dynamically Invoking Subprogram with Assoc. Array Formal Parameter	8-6
8-5	Dynamically Invoking Subprogram with Nested Table Formal Parameter	8-7
8-6	Dynamically Invoking Subprogram with Varray Formal Parameter	8-7
8-7	Uninitialized Variable Represents NULL in USING Clause	8-8
8-8	Native Dynamic SQL with OPEN FOR, FETCH, and CLOSE Statements	8-9
8-9	Querying a Collection with Native Dynamic SQL	8-9
8-10	Repeated Placeholder Names in Dynamic PL/SQL Block	8-11
8-11	DBMS_SQL.RETURN_RESULT Procedure	8-13
8-12	DBMS_SQL.GET_NEXT_RESULT Procedure	8-14
8-13	Switching from DBMS_SQL Package to Native Dynamic SQL	8-16
8-14	Switching from Native Dynamic SQL to DBMS_SQL Package	8-17
8-15	Setup for SQL Injection Examples	8-18
8-16	Procedure Vulnerable to Statement Modification	8-19
8-17	Procedure Vulnerable to Statement Injection	8-20
8-18	Procedure Vulnerable to SQL Injection Through Data Type Conversion	8-22
8-19	Bind Variables Guarding Against SQL Injection	8-24
8-20	Validation Checks Guarding Against SQL Injection	8-26
8-21	Explicit Format Models Guarding Against SQL Injection	8-27
9-1	Declaring, Defining, and Invoking a Simple PL/SQL Procedure	9-4

9-2	Declaring, Defining, and Invoking a Simple PL/SQL Function	9-5
9-3	Execution Resumes After RETURN Statement in Function	9-7
9-4	Function Where Not Every Execution Path Leads to RETURN Statement	9-7
9-5	Function Where Every Execution Path Leads to RETURN Statement	9-7
9-6	Execution Resumes After RETURN Statement in Procedure	9-8
9-7	Execution Resumes After RETURN Statement in Anonymous Block	9-8
9-8	Nested Subprograms Invoke Each Other	9-9
9-9	Formal Parameters and Actual Parameters	9-11
9-10	Actual Parameter Inherits Only NOT NULL from Subtype	9-12
9-11	Actual Parameter and Return Value Inherit Only Range From Subtype	9-13
9-12	Function Implicitly Converts Formal Parameter to Constrained Subtype	9-13
9-13	Avoiding Implicit Conversion of Actual Parameters	9-14
9-14	Parameter Values Before, During, and After Procedure Invocation	9-16
9-15	OUT and IN OUT Parameter Values After Exception Handling	9-18
9-16	OUT Formal Parameter of Record Type with Non-NULL Default Value	9-19
9-17	Aliasing from Global Variable as Actual Parameter	9-21
9-18	Aliasing from Same Actual Parameter for Multiple Formal Parameters	9-21
9-19	Aliasing from Cursor Variable Subprogram Parameters	9-22
9-20	Procedure with Default Parameter Values	9-23
9-21	Function Provides Default Parameter Value	9-23
9-22	Adding Subprogram Parameter Without Changing Existing Invocations	9-24
9-23	Equivalent Invocations with Different Notations in Anonymous Block	9-26
9-24	Equivalent Invocations with Different Notations in SELECT Statements	9-27
9-25	Resolving PL/SQL Procedure Names	9-28
9-26	Overloaded Subprogram	9-30
9-27	Overload Error Causes Compile-Time Error	9-33
9-28	Overload Error Compiles Successfully	9-33
9-29	Invoking Subprogram in Causes Compile-Time Error	9-33
9-30	Correcting Overload Error in	9-33
9-31	Invoking Subprogram in	9-33
9-32	Package Specification Without Overload Errors	9-34
9-33	Improper Invocation of Properly Overloaded Subprogram	9-34
9-34	Implicit Conversion of Parameters Causes Overload Error	9-34
9-35	Implicit Conversion to Number Successful	9-34
9-36	Implicit Conversion to BOOLEAN or Number Causes Overload Error	9-35
9-37	Recursive Function Returns n Factorial (n!)	9-36
9-38	Recursive Function Returns nth Fibonacci Number	9-36

9-39	Declaring and Defining Result-Cached Function	9-39
9-40	Result-Cached Function Returns Configuration Parameter Setting	9-42
9-41	Result-Cached Function Handles Session-Specific Settings	9-45
9-42	Result-Cached Function Handles Session-Specific Application Context	9-46
9-43	Caching One Name at a Time (Finer Granularity)	9-47
9-44	Caching Translated Names One Language at a Time (Coarser Granularity)	9-47
9-45	Database Link in a DR Unit	9-55
9-46	PL/SQL Anonymous Block Invokes External Procedure	9-56
9-47	PL/SQL Standalone Procedure Invokes External Procedure	9-56
9-48	Implement JavaScript External Procedure	9-57
10-1	Trigger Uses Conditional Predicates to Detect Triggering Statement	10-5
10-2	INSTEAD OF Trigger	10-6
10-3	INSTEAD OF Trigger on Nested Table Column of View	10-8
10-4	Compound Trigger Logs Changes to One Table in Another Table	10-12
10-5	Compound Trigger Avoids Mutating-Table Error	10-15
10-6	Foreign Key Trigger for Child Table	10-17
10-7	UPDATE and DELETE RESTRICT Trigger for Parent Table	10-18
10-8	UPDATE and DELETE SET NULL Trigger for Parent Table	10-19
10-9	DELETE CASCADE Trigger for Parent Table	10-20
10-10	UPDATE CASCADE Trigger for Parent Table	10-20
10-11	Trigger Checks Complex Constraints	10-22
10-12	Trigger Enforces Security Authorizations	10-23
10-13	Trigger Derives New Column Values	10-24
10-14	Trigger Logs Changes to EMPLOYEES.SALARY	10-29
10-15	Conditional Trigger Prints Salary Change Information	10-30
10-16	Trigger Modifies CLOB Columns	10-32
10-17	Trigger with REFERENCING Clause	10-32
10-18	Trigger References OBJECT_VALUE Pseudocolumn	10-33
10-19	BEFORE Statement Trigger on Sample Schema HR	10-35
10-20	AFTER Statement Trigger on Database	10-35
10-21	Trigger Monitors Logons	10-36
10-22	INSTEAD OF CREATE Trigger on Schema	10-36
10-23	Trigger Invokes Java Subprogram	10-36
10-24	Trigger Cannot Handle Exception if Remote Database is Unavailable	10-39
10-25	Workaround for	10-39
10-26	Trigger Causes Mutating-Table Error	10-43
10-27	Update Cascade	10-44

10-28	Viewing Information About Triggers	10-60
11-1	Simple Package Specification	11-4
11-2	Passing Associative Array to Standalone Subprogram	11-5
11-3	Matching Package Specification and Body	11-6
11-4	Creating SERIALY_REUSEABLE Packages	11-9
11-5	Effect of SERIALY_REUSEABLE Pragma	11-10
11-6	Cursor in SERIALY_REUSEABLE Package Open at Call Boundary	11-10
11-7	Separating Cursor Declaration and Definition in Package	11-12
11-8	ACCESSIBLE BY Clause	11-13
11-9	Creating emp_admin Package	11-14
12-1	Setting Value of PLSQL_WARNINGS Compilation Parameter	12-3
12-2	Displaying and Setting PLSQL_WARNINGS with DBMS_WARNING Subprograms	12-4
12-3	Single Exception Handler for Multiple Exceptions	12-7
12-4	Locator Variables for Statements that Share Exception Handler	12-8
12-5	Naming Internally Defined Exception	12-10
12-6	Anonymous Block Handles ZERO_DIVIDE	12-12
12-7	Anonymous Block Avoids ZERO_DIVIDE	12-12
12-8	Anonymous Block Handles ROWTYPE_MISMATCH	12-12
12-9	Redeclared Predefined Identifier	12-14
12-10	Declaring, Raising, and Handling User-Defined Exception	12-16
12-11	Explicitly Raising Predefined Exception	12-16
12-12	Reraising Exception	12-17
12-13	Raising User-Defined Exception with RAISE_APPLICATION_ERROR	12-18
12-14	Exception that Propagates Beyond Scope is Handled	12-21
12-15	Exception that Propagates Beyond Scope is Not Handled	12-21
12-16	Exception Raised in Declaration is Not Handled	12-22
12-17	Exception Raised in Declaration is Handled by Enclosing Block	12-22
12-18	Exception Raised in Exception Handler is Not Handled	12-23
12-19	Exception Raised in Exception Handler is Handled by Invoker	12-24
12-20	Exception Raised in Exception Handler is Handled by Enclosing Block	12-24
12-21	Exception Raised in Exception Handler is Not Handled	12-25
12-22	Exception Raised in Exception Handler is Handled by Enclosing Block	12-26
12-23	Displaying SQLCODE and SQLERRM Values	12-28
12-24	Exception Handler Runs and Execution Ends	12-29
12-25	Exception Handler Runs and Execution Continues	12-29
12-26	Retrying Transaction After Handling Exception	12-30
13-1	Specifying that Subprogram Is To Be Inlined	13-3

13-2	Specifying that Overloaded Subprogram Is To Be Inlined	13-3
13-3	Specifying that Subprogram Is Not To Be Inlined	13-4
13-4	PRAGMA INLINE ... 'NO' Overrides PRAGMA INLINE ... 'YES'	13-4
13-5	Nested Query Improves Performance	13-6
13-6	NOCOPY Subprogram Parameters	13-8
13-7	DELETE Statement in FOR LOOP Statement	13-13
13-8	DELETE Statement in FORALL Statement	13-14
13-9	Time Difference for INSERT Statement in FOR LOOP and FORALL Statements	13-14
13-10	FORALL Statement for Subset of Collection	13-15
13-11	FORALL Statements for Sparse Collection and Its Subsets	13-16
13-12	Handling FORALL Exceptions Immediately	13-19
13-13	Handling FORALL Exceptions After FORALL Statement Completes	13-21
13-14	Showing Number of Rows Affected by Each DELETE in FORALL	13-23
13-15	Showing Number of Rows Affected by Each INSERT SELECT in FORALL	13-24
13-16	Bulk-Selecting Two Database Columns into Two Nested Tables	13-26
13-17	Bulk-Selecting into Nested Table of Records	13-27
13-18	SELECT BULK COLLECT INTO Statement with Unexpected Results	13-28
13-19	Cursor Workaround for	13-29
13-20	Second Collection Workaround for	13-30
13-21	Limiting Bulk Selection with ROWNUM, SAMPLE, and FETCH FIRST	13-33
13-22	Bulk-Fetching into Two Nested Tables	13-33
13-23	Bulk-Fetching into Nested Table of Records	13-36
13-24	Limiting Bulk FETCH with LIMIT	13-37
13-25	Returning Deleted Rows in Two Nested Tables	13-38
13-26	Returning NEW and OLD Values of Updated Rows	13-39
13-27	DELETE with RETURN BULK COLLECT INTO in FORALL Statement	13-39
13-28	DELETE with RETURN BULK COLLECT INTO in FOR LOOP Statement	13-40
13-29	Anonymous Block Bulk-Binds Input Host Array	13-41
13-30	Creating and Invoking Pipelined Table Function	13-45
13-31	Pipelined Table Function Transforms Each Row to Two Rows	13-46
13-32	Fetching from Results of Pipelined Table Functions	13-48
13-33	Pipelined Table Function with Two Cursor Variable Parameters	13-49
13-34	Pipelined Table Function as Aggregate Function	13-50
13-35	Pipelined Table Function Does Not Handle NO_DATA_NEEDED	13-52
13-36	Pipelined Table Function Handles NO_DATA_NEEDED	13-53
13-37	Skip_col Polymorphic Table Function Example	13-58
13-38	To_doc Polymorphic Table Function Example	13-62

13-39	Implicit_echo Polymorphic Table Function Example	13-65
14-1	Restricting Access to Top-Level Procedures in the Same Schema	14-5
14-2	Restricting Access to a Unit Name of Any Kind	14-6
14-3	Restricting Access to a Stored Procedure	14-7
14-4	Nested, Labeled Basic LOOP Statements with EXIT WHEN Statements	14-14
14-5	Nested, Unabeled Basic LOOP Statements with EXIT WHEN Statements	14-14
14-6	External Function Example	14-27
14-7	CONTINUE Statement in Basic LOOP Statement	14-45
14-8	CONTINUE WHEN Statement in Basic LOOP Statement	14-45
14-9	Marking a Single Basic Block as Infeasible to Test for Coverage	14-47
14-10	Marking a Line Range as Infeasible to Test for Coverage	14-48
14-11	Marking Entire Units or Individual Subprograms as Infeasible to Test for Coverage	14-48
14-12	Marking Internal Subprogram as Infeasible to Test for Coverage	14-49
14-13	Enabling the Deprecation Warnings	14-59
14-14	Deprecation of a PL/SQL Package	14-60
14-15	Deprecation of a PL/SQL Package with a Custom Warning	14-60
14-16	Deprecation of a PL/SQL Procedure	14-61
14-17	Deprecation of an Overloaded Procedure	14-61
14-18	Deprecation of a Constant and of an Exception	14-61
14-19	Using Conditional Compilation to Deprecate Entities in Some Database Releases	14-62
14-20	Deprecation of an Object Type	14-62
14-21	Deprecation of a Member Function in an Object Type Specification	14-63
14-22	Deprecation of Inherited Object Types	14-63
14-23	Deprecation Only Applies to Top Level Subprogram	14-65
14-24	Misplaced DEPRECATE Pragma	14-65
14-25	Mismatch of the Element Name and the DEPRECATE Pragma Argument	14-66
14-26	Basic LOOP Statement with EXIT Statement	14-83
14-27	Basic LOOP Statement with EXIT WHEN Statement	14-83
14-28	EXIT WHEN Statement in FOR LOOP Statement	14-101
14-29	EXIT WHEN Statement in Inner FOR LOOP Statement	14-101
14-30	CONTINUE WHEN Statement in Inner FOR LOOP Statement	14-102
14-31	GOTO Statement	14-111
14-32	Incorrect Label Placement	14-111
14-33	GOTO Statement Goes to Labeled NULL Statement	14-112
14-34	GOTO Statement Transfers Control to Enclosing Block	14-112
14-35	GOTO Statement Cannot Transfer Control into IF Statement	14-113
14-36	Emp_doc: Using a Scalar Macro to Convert Columns into a JSON or XML Document	14-170

14-37	Env: Using a Scalar Macro in a Scalar Expression	14-173
14-38	Budget : Using a Table Macro in a Table Expression	14-174
14-39	Take: Using a Table Macro with a Polymorphic View	14-175
14-40	Range : Using a Table Macro in a Table Expression	14-175
14-41	Enabling the PLW-6009 Warning	14-182
14-42	SUPPRESSES_WARNING_6009 Pragma in a Procedure	14-182
14-43	SUPPRESSES_WARNING_6009 Pragma in a Function	14-183
14-44	SUPPRESSES_WARNING_6009 Pragma in an Overloaded Subprogram in a Package Specification	14-183
14-45	SUPPRESSES_WARNING_6009 Pragma in a Forward Declaration in a Package Body	14-184
14-46	SUPPRESSES_WARNING_6009 Pragma in Object Type Methods	14-185
14-47	WHILE LOOP Statements	14-190
15-1	Recompiling a Function	15-4
15-2	Recompiling a Library	15-5
15-3	Recompiling a Package	15-7
15-4	Recompiling a Procedure	15-10
15-5	Disabling Triggers	15-12
15-6	Enabling Triggers	15-12
15-7	Adding a Member Function	15-21
15-8	Adding a Collection Attribute	15-21
15-9	Increasing the Number of Elements of a Collection Type	15-22
15-10	Increasing the Length of a Collection Type	15-22
15-11	Recompiling a Type	15-22
15-12	Recompiling a Type Specification	15-22
15-13	Evolving and Resetting an ADT	15-22
15-14	Creating a Function	15-28
15-15	Creating Aggregate Functions	15-29
15-16	Package Procedure in a Function	15-29
15-17	Creating Functions Using MLE Module and Inline Call Specifications	15-29
15-18	Creating a Library	15-34
15-19	Specifying an External Procedure Agent	15-34
15-20	Creating the Specification for the emp_mgmt Package	15-38
15-21	Creating the emp_mgmt Package Body	15-41
15-22	Creating a Procedure	15-45
15-23	Creating an External Procedure	15-46
15-24	Creating Procedures Using MLE Module and Inline Call Specifications	15-46
15-25	ADT Examples	15-74

15-26	Creating a Subtype	15-75
15-27	Creating a Type Hierarchy	15-75
15-28	Creating a Varray Type	15-76
15-29	Creating a Non-Persistable Nested Array	15-76
15-30	Creating a Non-Persistable Object Type	15-76
15-31	Creating a Non-Persistable Varray	15-76
15-32	Creating a Nested Table Type	15-77
15-33	Creating a Nested Table Type Containing a Varray	15-77
15-34	Constructor Example	15-77
15-35	Creating a Member Method	15-77
15-36	Creating a Static Method	15-78
15-37	Dropping a Function	15-85
15-38	Dropping a Library	15-87
15-39	Dropping a Package	15-88
15-40	Dropping a Procedure	15-90
15-41	Dropping a Trigger	15-91
15-42	Dropping an ADT	15-93
15-43	Dropping an ADT Body	15-94
A-1	SQL File with Two Wrappable PL/SQL Units	A-3
A-2	Wrapping File with PL/SQL Wrapper Utility	A-4
A-3	Running Wrapped File and Viewing Wrapped PL/SQL Units	A-5
A-4	Creating Wrapped Package Body with CREATE_WRAPPED Procedure	A-9
A-5	Viewing Package with Wrapped Body and Invoking Package Procedure	A-10
B-1	Qualified Names	B-2
B-2	Variable Name Interpreted as Column Name Causes Unintended Result	B-3
B-3	Fixing with Different Variable Name	B-4
B-4	Fixing with Block Label	B-4
B-5	Subprogram Name for Name Resolution	B-4
B-6	Inner Capture of Column Reference	B-7
B-7	Inner Capture of Attribute Reference	B-8
B-8	Qualifying ADT Attribute References	B-9
B-9	Qualifying References to Row Expressions	B-10

List of Figures

2-1	PL/SQL Engine	2-10
6-1	Varray of Maximum Size 10 with 7 Elements	6-11
6-2	Array and Nested Table	6-16
7-1	Transaction Control Flow	7-54
9-1	How PL/SQL Compiler Resolves Invocations	9-28
12-1	Exception Does Not Propagate	12-19
12-2	Exception Propagates from Inner Block to Outer Block	12-20
12-3	PL/SQL Returns Unhandled Exception Error to Host Environment	12-20

List of Tables

2-1	PL/SQL I/O-Processing Packages	2-6
2-2	PL/SQL Compilation Parameters	2-11
3-1	Punctuation Characters in Every Database Character Set	3-2
3-2	PL/SQL Delimiters	3-4
3-3	Operator Precedence	3-29
3-4	Logical Truth Table	3-31
3-5	Relational Operators	3-37
3-6	Operators Allowed in Static Expressions	3-49
4-1	Data Types with Different Maximum Sizes in PL/SQL and SQL	4-2
4-2	Predefined PL/SQL BINARY_FLOAT and BINARY_DOUBLE Constants	4-3
4-3	Predefined Subtypes of PLS_INTEGER Data Type	4-28
6-1	PL/SQL Collection Types	6-2
6-2	Collection Methods	6-32
9-1	PL/SQL Subprogram Parameter Modes	9-14
9-2	PL/SQL Subprogram Parameter Modes Characteristics	9-15
9-3	PL/SQL Actual Parameter Notations	9-26
9-4	Finer and Coarser Caching Granularity	9-47
10-1	Conditional Predicates	10-5
10-2	Compound Trigger Timing-Point Sections	10-11
10-3	Constraints and Triggers for Ensuring Referential Integrity	10-16
10-4	OLD and NEW Pseudorecord Field Values	10-29
10-5	System-Defined Event Attributes	10-51
10-6	Database Event Triggers	10-55
10-7	Client Event Triggers	10-56
12-1	Compile-Time Warning Categories	12-2
12-2	Exception Categories	12-6
12-3	PL/SQL Predefined Exceptions	12-11
13-1	Profiling and Tracing Tools Summary	13-68
14-1	Iterand Implicit Type Defaults	14-124
14-2	Summary of Possible Sharing Attributes by Application Common Object Type	14-167
C-1	PL/SQL Compiler Limits	C-1
D-1	PL/SQL Reserved Words	D-1
D-2	PL/SQL Keywords	D-2

Preface

Oracle Database PL/SQL Language Reference describes and explains how to use PL/SQL, the Oracle procedural extension of SQL.

Topics

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)
- [Syntax Descriptions](#)

Audience

Oracle Database PL/SQL Language Reference is intended for anyone who is developing PL/SQL-based applications for either an Oracle Database or an Oracle TimesTen In-Memory Database, including:

- Programmers
- Systems analysts
- Project managers
- Database administrators

To use this document effectively, you need a working knowledge of:

- Oracle Database
- Structured Query Language (SQL)
- Basic programming concepts such as `IF-THEN` statements, loops, procedures, and functions

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Related Documents

For more information, see these documents in the Oracle Database documentation set:

- *Oracle Database SQL Language Reference*
- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database JSON Developer's Guide*
- *Oracle Database SODA for PL/SQL Developer's Guide*
- *Oracle Database Development Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database SecureFiles and Large Objects Developer's Guide*
- *Oracle Database Object-Relational Developer's Guide*
- *Oracle Database Concepts*
- *Oracle Database Performance Tuning Guide*
- *Oracle Database Sample Schemas*



See Also:

<https://www.oracle.com/database/technologies/appdev/plsql.html>

Conventions

This document uses these text conventions:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.
{A B C}	Choose either A, B, or C.

Also:

- `*_view` means all static data dictionary views whose names end with `view`. For example, `*_ERRORS` means `ALL_ERRORS`, `DBA_ERRORS`, and `USER_ERRORS`. For more information about any static data dictionary view, or about static dictionary views in general, see *Oracle Database Reference*.
- Table names not qualified with schema names are in the sample schema `HR`. For information about the sample schemas, see *Oracle Database Sample Schemas*.

Syntax Descriptions

Syntax descriptions are provided in this book for various SQL, PL/SQL, or other command-line constructs in graphic form or Backus Naur Form (BNF). See *Oracle Database SQL Language Reference* for information about how to interpret these descriptions.