7 Advanced PL/SQL Features

This chapter introduces the advanced PL/SQL features and refers to other chapters or documents for more information.

Topics:

- PL/SQL Data Types
- Dynamic SQL
- PL/SQL Optimize Level
- Compiling PL/SQL Units for Native Execution
- Exception Handling
- Conditional Compilation
- Bulk Binding

> **See Also:**
>
> - PL/SQL for Application Developers
> - *Oracle Database PL/SQL Language Reference* for a complete description of PL/SQL

7.1 PL/SQL Data Types

The PL/SQL data types include the SQL data types, additional scalar data types, and composite data types. You define the composite data types. You can also define subtypes of the scalar data types.

> **See Also:**
>
> PL/SQL Data Types

7.2 Dynamic SQL

Dynamic SQL is a programming methodology for generating and running SQL statements at runtime. It is useful when writing general-purpose and flexible programs like dynamic query systems, when writing programs that must run database definition language (DDL) statements, or when you do not know at compile time the full text of a SQL statement or the number or data types of its input and output variables.

> **See Also:**
>
> PL/SQL Dynamic SQL

7.3 PL/SQL Optimize Level

The PL/SQL optimize level determines how much the PL/SQL optimizer can rearrange code for better performance. This level is set with the compilation parameter `PLSQL_OPTIMIZE_LEVEL`.

> **See Also:**
>
> - *Oracle Database Reference*
> - PLSQL_OPTIMIZE_LEVEL Compilation Parameter

7.4 Compiling PL/SQL Units for Native Execution

You can usually speed up PL/SQL units by compiling them into native code (processor-dependent system code), which is stored in the SYSTEM tablespace.

> **See Also:**
>
> Compiling PL/SQL Units for Native Execution for more information about compiling PL/SQL units for native execution

7.5 Exception Handling

Exceptions (PL/SQL runtime errors) can arise from design faults, coding mistakes, hardware failures, and many other sources. You cannot anticipate all possible exceptions, but you can write exception handlers that let your program to continue to operate in their presence.

> **See Also:**
>
> *Oracle Database PL/SQL Language Reference*

7.6 Conditional Compilation

Conditional compilation lets you customize the functionality of a PL/SQL application without removing source text. For example, you can:

- Use new features with the latest database release and disable them when running the application in an older database release.

- Activate debugging or tracing statements in the development environment and hide them when running the application at a production site.

However:

- Oracle recommends against using conditional compilation to change the attribute structure of a type, which can cause dependent objects to "go out of sync" or dependent tables to become inaccessible.

  To change the attribute structure of a type, Oracle recommends using the `ALTER TYPE` statement, which propagates changes to dependent objects.
- Conditional compilation is subject to restrictions.

> **See Also:**
>
> *Oracle Database PL/SQL Language Reference*.
>
> *Oracle Database SQL Language Reference*

7.7 Bulk Binding

Bulk binding minimizes the performance overhead of the communication between PL/SQL and SQL, which can greatly improve performance.

> **See Also:**
>
> Overview of Bulk Binding.