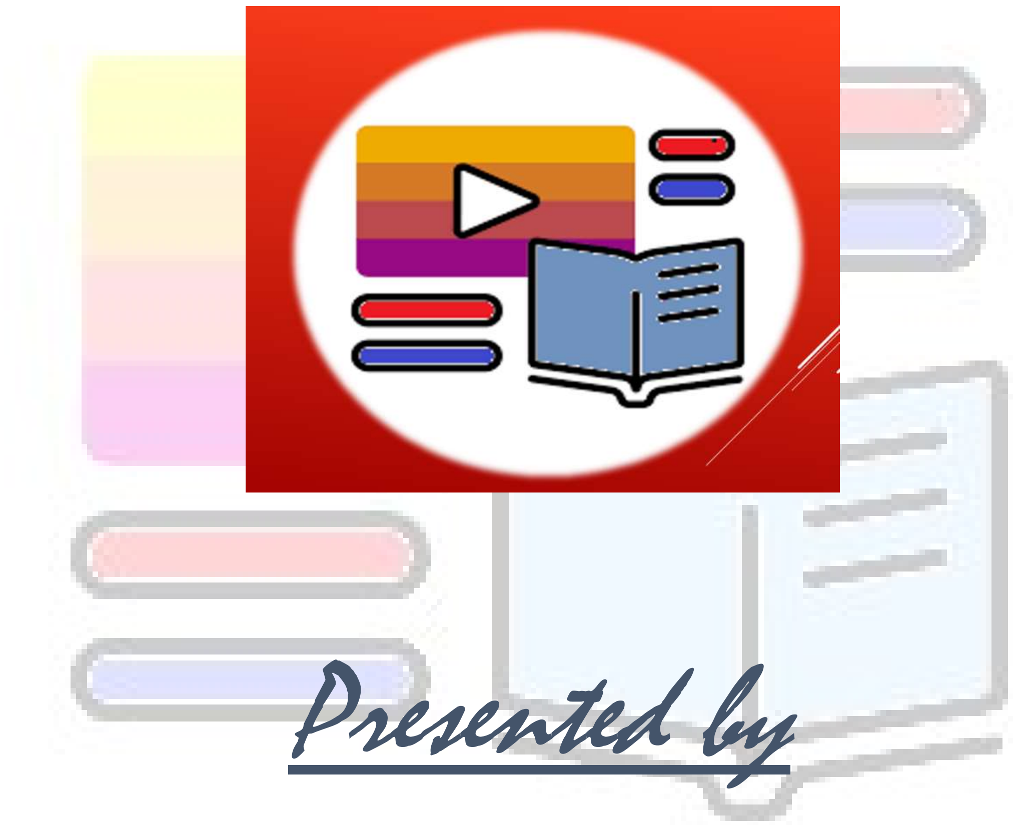
PL/SQL

(Procedural – Structured Query Language)

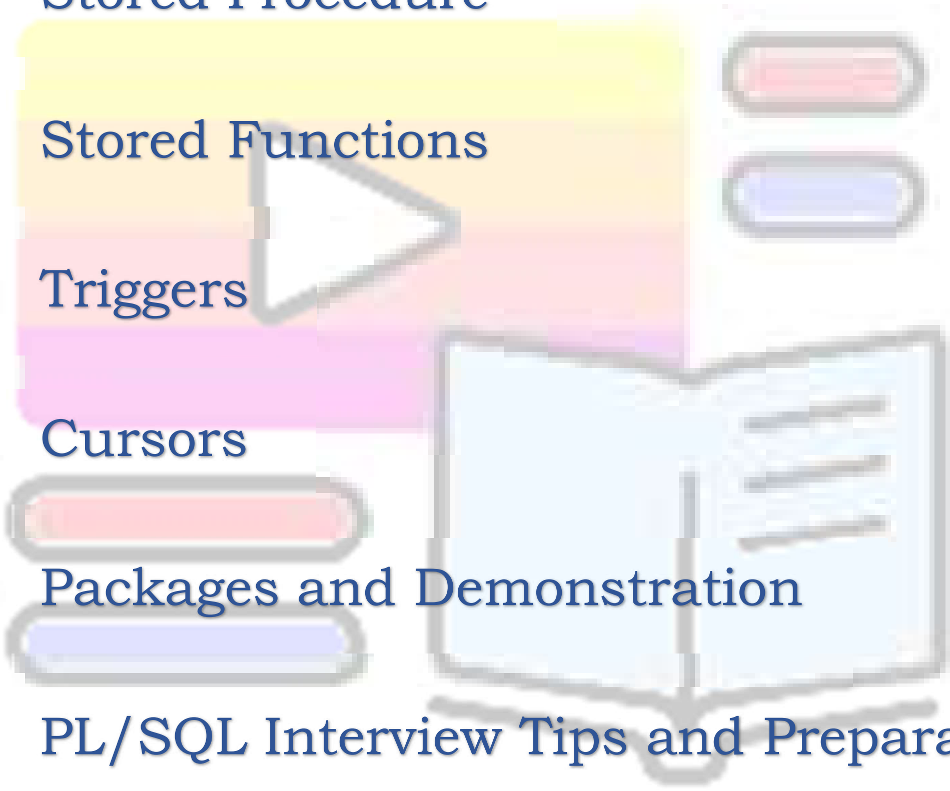


Presented by

TECH-RANCH

Let's make coding fun!

COURSE CONTENT



1. PL/SQL Introduction
2. Stored Procedure
3. Stored Functions
4. Triggers
5. Cursors
6. Packages and Demonstration
7. PL/SQL Interview Tips and Preparation

CHAPTER – 1

PL/ SQL INTRODUCTION

Let's make coding fun!

PL/SQL INTRODUCTION

PL/SQL is a combination of SQL along with the procedural features of programming languages. It was developed by Oracle Corporation in the early 90's to enhance the capabilities of SQL. PL/SQL is one of three key programming languages embedded in the Oracle Database, along with SQL itself and Java. This tutorial will give you great understanding on PL/SQL to proceed with Oracle database and other advanced RDBMS concepts. The PL/SQL programming language was developed by Oracle Corporation in the late 1980s as procedural extension language for SQL and the Oracle relational database. Following are certain notable facts about PL/SQL –

- PL/SQL is a completely portable, high-performance transaction-processing language.
- PL/SQL provides a built-in, interpreted and OS independent programming environment.
- PL/SQL can also directly be called from the command-line **SQL*Plus interface**.
- Direct call can also be made from external programming language calls to database.
- PL/SQL's general syntax is based on that of ADA and Pascal programming language.

Features of PL/SQL

- PL/SQL is tightly integrated with SQL.
- It offers extensive error checking.
- It offers numerous data types.
- It offers a variety of programming structures.
- It supports structured programming through functions and procedures.
- It supports object-oriented programming.
- It supports the development of web applications and server pages.

Advantages of PL/SQL over SQL

- SQL is the standard database language and PL/SQL is strongly integrated with SQL. PL/SQL supports both static and dynamic SQL. Static SQL supports DML operations and transaction control from PL/SQL block. In Dynamic SQL, SQL allows embedding DDL statements in PL/SQL blocks.
- PL/SQL allows sending an entire block of statements to the database at one time. This reduces network traffic and provides high performance for the applications.
- PL/SQL gives high productivity to programmers as it can query, transform, and update data in a database.
- PL/SQL saves time on design and debugging by strong features, such as exception handling, encapsulation, data hiding, and object-oriented data types.
- Applications written in PL/SQL are fully portable.
- PL/SQL provides high security level.
- PL/SQL provides access to predefined SQL packages.
- PL/SQL provides support for Object-Oriented Programming.
- PL/SQL provides support for developing Web Applications and Server Pages.

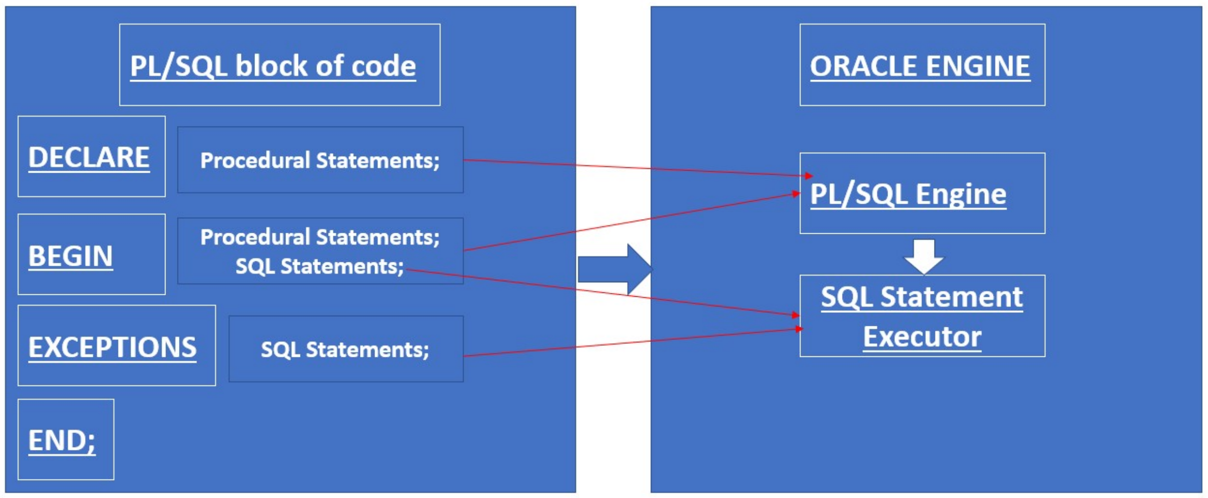
Declarations:- This section starts with the keyword **DECLARE**. It is an optional section and defines all variables, cursors, subprograms, and other elements to be used in the program.

Executable Commands :-This section is enclosed between the keywords **BEGIN** and **END** and it is a mandatory section. It consists of the executable PL/SQL statements of the program. It should have at least one executable line of code, which may be just a **NULL command** to indicate that nothing should be executed.

Exception Handling This section starts with the keyword **EXCEPTION**. This optional section contains **exception(s)** that handle errors in the program.

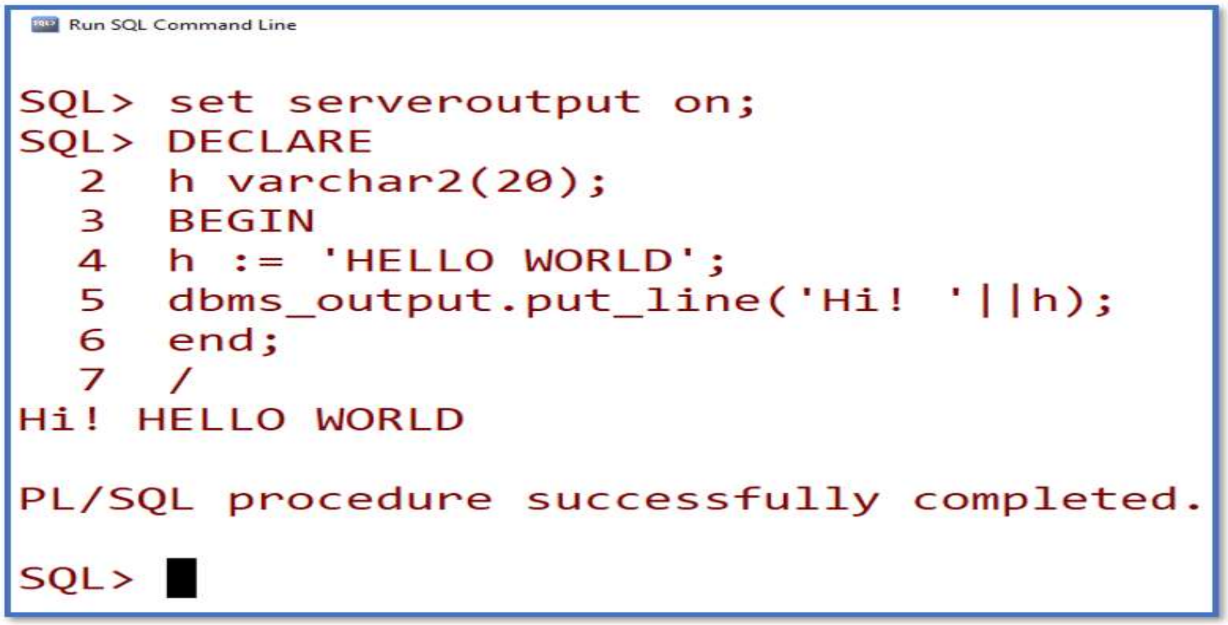
Every PL/SQL statement ends with a semicolon (;). PL/SQL blocks can be nested within other PL/SQL blocks using **BEGIN** and **END**.

PL/SQL Block



The 'Hello World' Example

HelloWorld PL/SQL Program



```
SQL> set serveroutput on;
SQL> DECLARE
  2  h varchar2(20);
  3  BEGIN
  4  h := 'HELLO WORLD';
  5  dbms_output.put_line('Hi! '||h);
  6  end;
  7  /
Hi! HELLO WORLD

PL/SQL procedure successfully completed.

SQL>
```

The **end;** line signals the end of the PL/SQL block. To run the code from the SQL command line, you may need to type / at the beginning of the first blank line after the last line of the code. When the above code is executed at the SQL prompt, it produces the following result −Hello World

The PL/SQL Identifiers

PL/SQL identifiers are constants, variables, exceptions, procedures, cursors, and reserved words. The identifiers consist of a letter optionally followed by more letters, numerals, dollar signs, underscores, and number signs and should not exceed 30 characters.

By default, **identifiers are not case-sensitive**. So you can use **integer** or **INTEGER** to represent a numeric value. You cannot use a reserved keyword as an identifier.

The PL/SQL Delimiters

A delimiter is a symbol with a special meaning. Following is the list of delimiters in PL/SQL −

Delimiter	Description
+, -, *, /	Addition, subtraction/negation, multiplication, division
%	Attribute indicator
'	Character string delimiter
.	Component selector
(,)	Expression or list delimiter
:	Host variable indicator
,	Item separator
"	Quoted identifier delimiter
=	Relational operator
@	Remote access indicator
;	Statement terminator

:=	Assignment operator
=>	Association operator
||	Concatenation operator
******	Exponentiation operator
<<, >>	Label delimiter (begin and end)
/*, */	Multi-line comment delimiter (begin and end)
--	Single-line comment indicator
..	Range operator
<, >, <=, >=	Relational operators
<>, '=, ~=, ^=	Different versions of NOT EQUAL

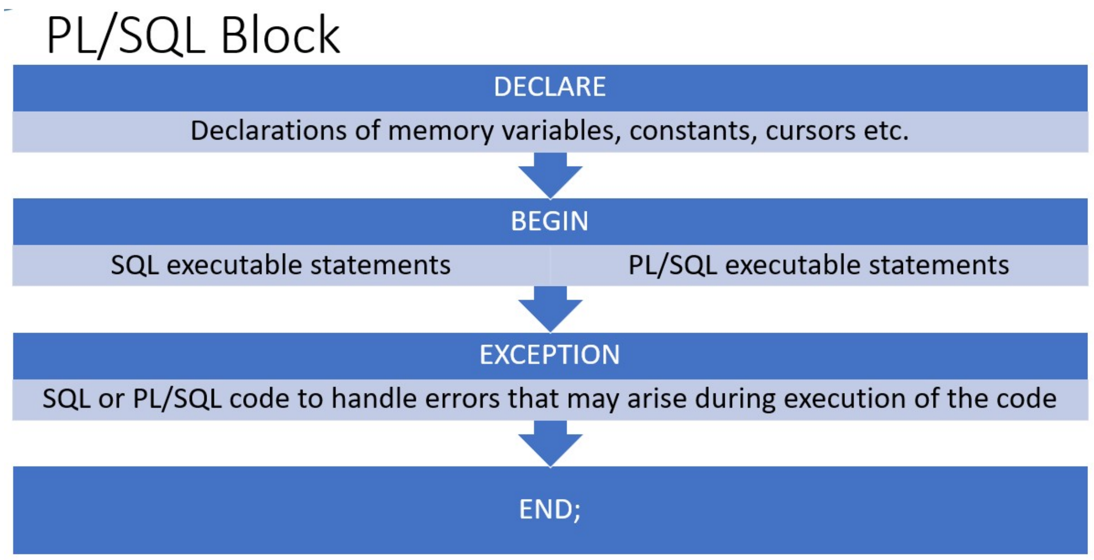
PL/SQL Program Units

A PL/SQL unit is any one of the following − PL/SQL block

- Function
- Package
- Package body
- Procedure
- Trigger

PL/SQL Scalar Data Types and Subtypes

Numeric	Numeric values on which arithmetic operations are performed.
Character	Alphanumeric values that represent single characters or strings of characters.
Boolean	Logical values on which logical operations are performed.
Datetime	Dates and times.


PL/SQL Block
DECLARE
Declarations of memory variables, constants, cursors etc.
BEGIN
SQL executable statements
PL/SQL executable statements
EXCEPTION
SQL or PL/SQL code to handle errors that may arise during execution of the code
END;

CONCLUSION

In this chapter, we have learnt about how to program PL/SQL , showed HelloWorld PL/SQL Program and its advantages over SQL, Why do we need PL/SQL?

1. Procedural capabilities
2. Fast data processing for Multi-user environment
3. Developmental tool supports data manipulation and facilities of conditional checking, branching and looping
4. Improves transaction performance
5. Portable