Introduction to PL/SQL

* PL/SQL stands for “Procedural Language extensions to the Structured Query Language”.
* [SQL](https://www.oracletutorial.com/oracle-basics/) is a popular language for both [querying](https://www.oracletutorial.com/oracle-basics/oracle-select/) and [updating data](https://www.oracletutorial.com/oracle-basics/oracle-update/) in the relational database management systems (RDBMS).
* PL/SQL adds many procedural constructs to SQL language to overcome some limitations of SQL like IF statements, Looping constructs etc.
* PL/SQL is not a stand-alone programming language; it is a tool within the Oracle programming environment.
* SQL\*Plus is an interactive tool that allows you to type SQL and PL/SQL statements at the command prompt.

Disadvantages of SQL:

* SQL doesn’t provide the programmers with a technique of condition checking, looping and branching.
* SQL statements are passed to Oracle engine one at a time which increases traffic and decreases speed.
* SQL has no facility of error checking during manipulation of data.

Features of PL/SQL:

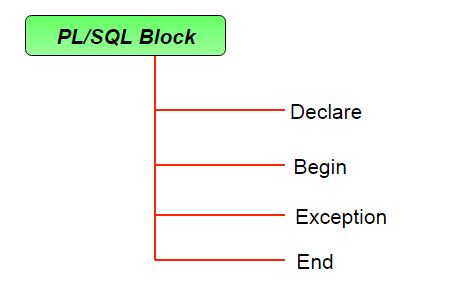
1. PL/SQL is basically a procedural language, which provides the functionality of decision making, iteration and many more features of procedural programming languages.
2. PL/SQL can execute a number of queries in one block using single command.
3. One can create a PL/SQL unit such as procedures, functions, packages, triggers, and types, which are stored in the database for reuse by applications.
4. PL/SQL provides a feature to handle the exception which occurs in PL/SQL block known as exception handling block.
5. Applications written in PL/SQL are portable to computer hardware or operating system where Oracle is operational.
6. PL/SQL Offers extensive error checking.
7. PL/SQL provides high security level.
8. PL/SQL provides access to predefined SQL packages.
9. PL/SQL provides support for Object-Oriented Programming.
10. PL/SQL provides support for developing Web Applications and Server Pages.

**Differences between SQL and PL/SQL:**

|  |  |
| --- | --- |
| **SQL** | **PL/SQL** |
| SQL is a single query that is used to perform DML and DDL operations. | PL/SQL is a block of codes that used to write the entire program blocks/ procedure/ function, etc. |
| It is declarative, that defines what needs to be done, rather than how things need to be done. | PL/SQL is procedural that defines how the things needs to be done. |
| Execute as a single statement. | Execute as a whole block. |
| Mainly used to manipulate data. | Mainly used to create an application. |
| Cannot contain PL/SQL code in it. | It is an extension of SQL, so it can contain SQL inside it. |

Structure of PL/SQL Block:

The basic unit in PL/SQL is a block. All PL/SQL programs are made up of blocks, which can be nested within each other.



Typically, each block performs a logical action in the program. A block has the following structure:

**DECLARE**

declaration statements; (optional)

**BEGIN**

executable statements

**EXCEPTIONS**

exception handling statements

**END;**

* Declare section starts with **DECLARE** keyword in which variables, constants, records as cursors can be declared which stores data temporarily. It basically consists definition of PL/SQL identifiers. This part of the code is optional.
* Execution section starts with **BEGIN** and ends with **END** keyword.This is a mandatory section and here the program logic is written to perform any task like loops and conditional statements. It supports all [DML](https://en.wikipedia.org/wiki/Data_manipulation_language) commands, [DDL](https://en.wikipedia.org/wiki/Data_definition_language) commands and SQL\*PLUS built-in functions as well.
* Exception section starts with **EXCEPTION** keyword.This section is optional which contains statements that are executed when a run-time error occurs. Any exceptions can be handled in this section.

### PL/SQL identifiers

There are several PL/SQL identifiers such as variables, constants, procedures, cursors, triggers etc.

1. **Variables**:  
   Like several other programming languages, variables in PL/SQL must be declared prior to its use. They should have a valid name and data type as well.

Syntax for declaration of variables:

variable\_name datatype [NOT NULL := value ];

Example to show how to declare variables in PL/SQL :

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

    var1 INTEGER;

    var2 REAL;

    var3 varchar2(20) ;

BEGIN

    null;

END;

/

* + **SET SERVEROUTPUT ON**: It is used to display the buffer used by the dbms\_output.
  + **Slash (/) after END;:** The slash (/) tells the SQL\*Plus to execute the block.

**1.1) INITIALISING VARIABLES:**  
The variables can also be initialised just like in other programming languages. Let us see an example for the same:

|  |
| --- |
| SQL> SET SERVEROUTPUT ON;  SQL> DECLARE       var1 INTEGER **:=** 2 ;       var3 varchar2(20) := 'Hello World' ;      BEGIN       null;      END;    / |

**Assignment operator (:=)** : It is used to assign a value to a variable.

1. **Displaying Output**:  
   The outputs are displayed by using DBMS\_OUTPUT which is a built-in package that enables the user to display output, debugging information, and send messages from PL/SQL blocks, subprograms, packages, and triggers.

Let us see an example to see how to display a message using PL/SQL :

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

     var varchar2(40) := 'I love GeeksForGeeks' ;

  BEGIN

     dbms\_output.put\_line(var);

  END;

  /

Output:

I love GeeksForGeeks

PL/SQL procedure successfully completed.

**Explanation:**

* + dbms\_output.put\_line : This command is used to direct the PL/SQL output to a screen.

1. **Using Comments**:  
   Like in many other programming languages, in PL/SQL also, comments can be put within the code which has no effect in the code. There are two syntaxes to create comments in PL/SQL :
   * **Single Line Comment:** To create a single line comment , the symbol – – is used.
   * **Multi Line Comment:** To create comments that span over several lines, the symbol /\* and \*/ is used.
2. **Taking input from user**:  
   Just like in other programming languages, in PL/SQL also, we can take input from the user and store it in a variable. Let us see an example to show how to take input from users in PL/SQL:

SQL> SET SERVEROUTPUT ON;

SQL> DECLARE

      -- taking input for variable a

      a number := &a;

      -- taking input for variable b

      b varchar2(30) := &b;

  BEGIN

      null;

  END;

/

**PL/SQL Execution Environment:**

The PL/SQL engine resides in the Oracle engine.The Oracle engine can process not only single SQL statement but also block of many statements.The call to Oracle engine needs to be made only once to execute any number of SQL statements if these SQL statements are bundled inside a PL/SQL block.