### Data Types In PL-SQL ====>
===============================

* PL/SQL has two kinds of data types: scalar and composite.

* `Scalar`types are types that store single values such as Number, Boolean, Character, and Datetime.

* `Composite` types are types that store multiple values, for example, record & collection.

* PL/SQL divides the scalar data types into four families:
- Number
- Boolean
- Character
- Datetime

## Numeric Types ==>
====================

* The numeric data types represent real numbers, integers, and floating-point numbers.

* The data types for this are:
- `Number` :- Same as SQL data type Number.
- `PLS_INTEGER` :- Datatype is specific to PL/SQL. It represents signed 32 bits integers that range from -2,147,483,648 to 2,147,483,647.
- `Integer or Int` :- An integer type with maximum precision of 38 decimal digits.
- `Float or Double Precision` :- For storing decimal values Boolean .

## Boolean Types ==>
====================

* The BOOLEAN data type has three data values: `TRUE`, `FALSE`, and `NULL`.

* Boolean values are typically used in control flow structure such as IF-THEN, CASE, and loop statements like LOOP, FOR LOOP, and WHILE LOOP.

## Character Types ==>
======================

* Character types let you store alphanumeric data, represent words and text, and manipulate character strings.

* Following are character types supported by PL-SQL:
- CHAR
- VARCHAR
- VARCHAR2
- RAW
- LONG

## DateTimeTypes ==>
====================

* The datetime data types represent dates and timestamp.

* The two popular types in this category are:
- `DATE`
- `TIMESTAMP` :- The TIMESTAMP data type allows us to store date and time data including year, month, day, hour, minute and second.
 In addition, it stores the fractional seconds, which is not stored by the DATE data type.

## Two Special Types ==>
========================

* There are two special types also called %TYPE and %ROWTYPE.
- `%TYPE` :- is used to define the data type of variable as the column name datatype specified for a table.

- `%ROWTYPE` :- Used to declare a record with the same types as found in the specified table.

## Rules For Variable Declaration ===>
======================================

* Variable names cannot be longer than 30 characters.

* They must begin with an alphabetical character. Although we can have numbers and certain special characters in the name, but the first character must be an alpha character.

* They can contain only alphabetical characters, numbers, or one of the following special characters: # $ _

* Variable names are not case sensitive.

# Variable Declaration ==>
==========================

* PL/SQL variables must be declared in the declaration section with the following syntax:
* Syntax :-
# Variable_name datatype(size);

* Example :-
# roll_no NUMBER(2);
# a int;

# Variable Assignment ==>
=========================

* Whenever we declare a variable, PL/SQL assigns it a default value of NULL.

* If we want to initialize a variable with a value other than the NULL value, we can do so during the declaration, using either of the following –
- The DEFAULT keyword
- The assignment operator which is :=

* Example :-
# counter int:= 0;
# greetings varchar2(20) DEFAULT 'Have a Good Day';

# PL-SQL Constants ==>
======================

* Constants are those values which when declared remain fixed throughout the PL/SQL block.

* For declaring constants, a constant keyword is used.

* Syntax :-
# Constant_Name constant Datatype(size) :=<value>;

* Example :-
# school_name constant VARCHAR2(20) := “DPS";

# PL-SQL Comments ==>
=====================

* PL/SQL has two comment styles: single-line and multi-line comments.

* `Single Line Comments` :- A single-line comment starts with a double hyphen (--) that can appear anywhere on a line and extends to the end of the line.

- Example :-
# -- a:=10;

* `Multi Line Comments` :- A multi-line comment starts with a slash-asterisk (/*) and ends with an asterisk-slash (*/), and can span multiple lines:

- Example:
# /* This is a multi-line comment
#   that can span multiple lines */