**What is PL/SQL Packages?**

A package is a schema object that groups logically related PL/SQL types, variables, and subprograms. Packages usually have two parts, a specification (spec) and a body; sometimes the body is unnecessary. The specification is the interface to the package. It declares the types, variables, constants, exceptions, cursors, and subprograms that can be referenced from outside the package. The body defines the queries for the cursors and the code for the subprograms.  
To create package specs, use the [SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/introduction-to-sql/) statement CREATE PACKAGE. A CREATE PACKAGE BODY statement defines the package body.

***Enroll yourself in***[***PL/SQL Training***](https://intellipaat.com/pl-sql-online-training/)***and give a head-start to your career in PL?SQL!***

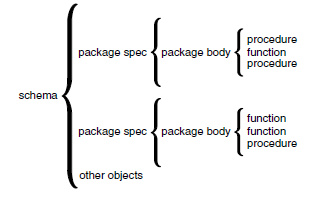
**What Goes In a PL/SQL Package?**  
The following is contained in a PL/SQL package:

* Get and Set methods for the package variables, if you want to avoid letting other procedures read and write them directly.
* Cursor declarations with the text of SQL queries. Reusing exactly the same query text in multiple locations is faster than retyping the same query each time with slight differences. It is also easier to maintain if you need to change a query that is used in many places.
* Declarations for exceptions. Typically, you need to be able to reference these from different procedures, so that you can handle exceptions within called subprograms.
* Declarations for procedures and functions that call each other. You do not need to worry about compilation order for packaged procedures and functions, making them more convenient than standalone stored procedures and functions when they call back and forth to each other.
* Declarations for overloaded procedures and functions. You can create multiple variations of a procedure or function, using the same names but different sets of parameters.
* Variables that you want to remain available between procedure calls in the same session. You can treat variables in a package like global variables.
* Type declarations for PL/SQL collection types. To pass a collection as a parameter between stored procedures or functions, you must declare the type in a package so that both the calling and called subprogram can refer to it.

***Come to Intellipaat’s***[***PL/SQL Community***](https://intellipaat.com/community/tag/plsql)***if you have more queries on PL/SQL!***

**Advantages of PL/SQL Packages**

* Modularity
* Easier Application Design
* Information Hiding
* Added Functionality
* Better Performance

[](https://cdn.intellipaat.com/wp-content/uploads/2015/09/package-scope.png)

**Example: A Simple Package Specification Without a Body**

CREATE PACKAGE trans\_data AS -- bodiless package

TYPE TimeRec IS RECORD (

minutes SMALLINT,

hours SMALLINT);

TYPE TransRec IS RECORD (

category VARCHAR2(10),

account INT,

amount REAL,

time\_of TimeRec);

minimum\_balance CONSTANT REAL := 10.00;

number\_processed INT;

insufficient\_funds EXCEPTION;

END trans\_data;

/

**The Package Body**  
The package body contains the implementation of every cursor and subprogram declared in the package spec. Subprograms defined in a package body are accessible outside the package only if their specs also appear in the package spec. If a subprogram spec is not included in the package spec, that subprogram can only be called by other subprograms in the same package.  
A package body must be in the same schema as the package spec. To match subprogram specs and bodies, PL/SQL does a token-by-token comparison of their headers. Except for white space, the headers must match word for word. Otherwise, PL/SQL raises an exception.  
**Example: Matching Package Specifications and Bodies**

CREATE PACKAGE emp\_bonus AS

PROCEDURE calc\_bonus (date\_hired employees.hire\_date%TYPE);

END emp\_bonus;

/

CREATE PACKAGE BODY emp\_bonus AS

-- the following parameter declaration raises an exception

-- because 'DATE' does not match employees.hire\_date%TYPE

-- PROCEDURE calc\_bonus (**date\_hired DATE**) IS

-- the following is correct because there is an exact match

PROCEDURE calc\_bonus (**date\_hired employees.hire\_date%TYPE**) IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Employees hired on ' || date\_hired || ' get bonus.');

END;

END emp\_bonus;

/

**How to Package STANDARD Defines the PL/SQL Environment**  
A package named STANDARD defines the PL/SQL environment. The package spec globally declares types, exceptions, and [PL/SQL subprograms](https://intellipaat.com/blog/tutorial/oracle-plsql-tutorial/plsql-subprograms/), which are available automatically to PL/SQL programs. For example, the package STANDARD declares function ABS, which returns the absolute value of its argument, as follows:

FUNCTION ABS (n NUMBER) RETURN NUMBER;

The contents of package STANDARD are directly visible to applications. You do not need to qualify references to its contents by prefixing the package name.

***Get familiar with the top***[***PL/SQL Interview Questions***](https://intellipaat.com/blog/interview-question/pl-sql-interview-questions/)***to get a head start in your career!***

**Product-Specific Packages**  
Oracle and various Oracle tools are supplied with product-specific packages that define application programming interfaces (APIs) you can call from PL/SQL, [SQL](https://intellipaat.com/blog/tutorial/sql-tutorial/), [Java](https://intellipaat.com/blog/tutorial/java-tutorial/), or other programming environments.

* **DBMS\_ALERT Package**

Package DBMS\_ALERT lets you use database triggers to alert an application when specific database values change.

* **DBMS\_OUTPUT Package**

Package DBMS\_OUTPUT enables you to display output from PL/SQL blocks, subprograms, packages, and triggers.

* **DBMS\_PIPE Package**

Package DBMS\_PIPE allows different sessions to communicate over named pipes. (A pipe is an area of memory used by one process to pass information to another.)

* **HTF and HTP Packages**

Packages HTF and HTP allow your PL/SQL programs to generate HTML tags.

* **UTL\_FILE Package**

Package UTL\_FILE lets PL/SQL programs read and write operating system (OS) text files. It provides a restricted version of standard OS stream file I/O, including open, put, get, and close operations.

* **UTL\_HTTP Package**

Package UTL\_HTTP allows your PL/SQL programs to make hypertext transfer protocol (HTTP) callouts.

* **UTL\_SMTP Package**

Package UTL\_SMTP allows your PL/SQL programs to send electronic mails (emails) over Simple Mail Transfer Protocol (SMTP)