Package

- All related function and procedure can be grouped together in a single unit called packages.
- Package is reliable to granting privileges.
- All function and procedure within a package can share variable among them.
- Package enables to perform "overloading" of functions and procedures.
- Package improve performance by loading the multiple object into memory at once, therefore, subsequent calls to related program do not required physical I/O.
- Package is reduce the traffic because all block execute all at once

Components of Packages

- **Specification**: It contains the list of various functions procedure names which will be a part of the package. Declare the type, variable, constant, exception cursor and subprogram
- **Body**: This contains the actual PL/SQL statement code implementing the logics of functions and procedures declared in "specification".
- Any items declared inside the package specification are visible outside the package.
- These objects declared in the package specification are called public.
- objects that are declared inside the package body, you are restricted to use within that package.(Private)
- variable, constant, or cursor was declared in a package specification or body, their values persist for the duration of the user's session.
- The values are lost when the current user's session terminates or the package is recompiled.

Defining Package Specification

CREATE or REPLACE PACKAGE <Package Name> {is,as} PROCEDURE

[Schema..] <ProcedureName>

```
            (<argument> {IN,OUT,IN OUT} <Data Type>,..);
  FUNCTION [Schema..]   <Function Name>
            (<argument> IN <Data Type>,..)
            RETURN <Data Type>);
```

Creating Package Body

CREATE or REPLACE PACKAGE BODY <Package Name> {is,as} PROCEDURE [Schema..] <ProcedureName>

```
          (<argument> {IN,OUT,IN OUT} <Data Type>,..) {IS, AS}
<variable> declarations;
<constant>
declarations; BEGIN
<PL/SQL subprogram
   body> EXCEPTION
<PL/SQL Exception
block> END;
FUNCTION [Schema..] <FunctionName>(<argument> IN <Data Type>,..) return <Data
Type> {IS,AS}
          <variable> declarations;
          <constant>
declarations; BEGIN
          <PL/SQL
subprogram body>
EXCEPTION
          <PL/SQL
Exception block> END;
END;
```

example

```
create or replace package math_pow
as
   procedure squ(n number);
   procedure cube(n number);
end;
```

```
/

create or replace package body math_pow
as
   procedure squ(n in number)
     is
       begin
         dbms_output.put_line('Square is '|| (n*n));
       end;

    procedure cube(n in number)
      is
       begin
         dbms_output.put_line('Cube is '|| (n*n*n));
       end;
end;
/

SQL> exec math_pow.squ(4);
Square is 16
```

Design a package which can accept employee used id and retrieve employee name ,salary(use procedure).using that salary calculate bonus (use function)

```
create or replace package emps

as

   procedure emp_name;

    function cal_bonus(s number) return number;

end;


create or replace package body emps

as

   procedure emp_name

   as

     e number;

     n varchar2(30);

     s number;

     ans number;

   Begin
```

```
    e:=&e;
     select ename,salary into n,s from emp where eid=e;
      ans:=cal_bonus(s);
     dbms_output.put_line('Emp Name is ' ||n);
     dbms_output.put_line('Emp salary is ' ||s);
     dbms_output.put_line('Emp bonus is ' ||ans);
  end emp_name;
 function cal_bonus( s in number)
  return number
   as
    b number;
  Begin
    if(s>=10000 and s<20000) then
        b:=s*0.30;
    elsif (s>=20000 and s<30000) then
        b:=s*0.20;
    end if;
    return b;
  end cal_bonus;
end;
/
```

Call procedure using package

```
exec emps.emp_name;
```

Public Objects

- **Public objects** are those that are declared in the **package specification** and are accessible to any program or user that has permission to access the package.
- These objects can be **procedures**, **functions**, **variables**, **cursors**, etc.
- The **public interface** is what external programs or users interact with. They can call public functions and procedures directly.

Private Objects

- **Private objects** are declared in the **package body** and are only accessible within the package itself (i.e., they are hidden from external users and programs).
- Private objects help in encapsulating the implementation details of the package, ensuring that they cannot be accessed directly from outside the package.
- These objects might include helper functions, internal variables, or data structures that are used to implement the logic of public objects.

```
   CREATE PACKAGE my_package
    AS
       PROCEDURE public_proc;
       FUNCTION public_func RETURN NUMBER;
END my_package;


CREATE PACKAGE BODY my_package AS
 PROCEDURE private_proc IS
   BEGIN
     Dbms_output.put_line('private procedure is call through procedure inside
                                         Package only');
    END private_proc;
 PROCEDURE public_proc
 IS
   BEGIN
            Dbms_output.put_line('Public procedure is call through package');
   END public_proc;
END my_package;
```

In above package private object is called within package.finally execute public procedure using package name

Exec my_package.public_proc

Private object is called inside public object