GALGOTIAS UNIVERSITY

School of Computing Science and Engineering

Program: BCA - IOP

Course Code: BCAS3031

Course Name: PL/SQL & Cursors and Triggers

Dr. T. Poongodi

Associate Professor

Packages are schema objects that groups logically related PL/SQL types, variables, and subprograms.

A package will have two mandatory parts –

- Package specification
- Package body or definition

Package Specification

- The specification is the interface to the package.
- It just DECLARES the types, variables, constants, exceptions, cursors, and subprograms that can be referenced from outside the package.
- In other words, it contains all information about the content of the package, but excludes the code for the subprograms.

- All objects placed in the specification are called public objects.
- Any subprogram not in the package specification but coded in the package body is called a private object.
- The following code snippet shows a package specification having a single procedure.
- You can have many global variables defined and multiple procedures or functions inside a package.

```
CREATE PACKAGE cust_sal AS
   PROCEDURE find_sal(c_id customers.id%type);
END cust_sal;
/
```

When the above code is executed at the SQL prompt, it produces the following result –

```
Package created.
```

Package Body

- The package body has the codes for various methods declared in the package specification and other private declarations, which are hidden from the code outside the package.
- The CREATE PACKAGE BODY Statement is used for creating the package body.
- The following code snippet shows the package body declaration for the cust_sal package.
- It is assumed that we already have CUSTOMERS table created in our database.

```
CREATE OR REPLACE PACKAGE BODY cust_sal AS

  PROCEDURE find_sal(c_id customers.id%TYPE) IS
  c_sal customers.salary%TYPE;
  BEGIN
    SELECT salary INTO c_sal
    FROM customers
    WHERE id = c_id;
    dbms_output.put_line('Salary: '|| c_sal);
  END find_sal;
END cust_sal;
/
```

Using the Package Elements

The package elements (variables, procedures or functions) are accessed with the following syntax –

package_name.element_name;

Consider, we already have created the above package in our database schema, the following program uses the find_sal method of the cust_sal package –

```
DECLARE
   code customers.id%type := &cc_id;
BEGIN
   cust_sal.find_sal(code);
END;
/
```

When the above code is executed at the SQL prompt, it prompts to enter the customer ID and when you enter an ID, it displays the corresponding salary as follows −

```
Enter value for cc_id: 1
Salary: 3000


PL/SQL procedure successfully completed.
```

Example

The following program provides a more complete package. We will use the CUSTOMERS table stored in our database with the following records –

```
Select * from customers;
```

```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  3000.00 |
|  2 | Khilan   |  25 | Delhi     |  3000.00 |
|  3 | kaushik  |  23 | Kota      |  3000.00 |
|  4 | Chaitali |  25 | Mumbai    |  7500.00 |
|  5 | Hardik   |  27 | Bhopal    |  9500.00 |
|  6 | Komal    |  22 | MP        |  5500.00 |
+----+----------+-----+-----------+----------+
```

The Package Specification

```
CREATE OR REPLACE PACKAGE c_package AS
  -- Adds a customer
  PROCEDURE addCustomer(c_id   customers.id%type,
  c_name  customers.name%type,
  c_age  customers.age%type,
  c_addr customers.address%type,
  c_sal  customers.salary%type);

  -- Removes a customer
  PROCEDURE delCustomer(c_id  customers.id%TYPE);
  --Lists all customers
  PROCEDURE listCustomer;

END c_package;
/
```

When the above code is executed at the SQL prompt, it creates the above package and displays the following result –

Package created.

Creating the Package Body

```
CREATE OR REPLACE PACKAGE BODY c_package AS
  PROCEDURE addCustomer(c_id  customers.id%type,
    c_name customers.name%type,
    c_age  customers.age%type,
    c_addr  customers.address%type,
    c_sal   customers.salary%type)
  IS
  BEGIN
    INSERT INTO customers (id,name,age,address,salary)
      VALUES(c_id, c_name, c_age, c_addr, c_sal);
  END addCustomer;

  PROCEDURE delCustomer(c_id   customers.id%type) IS
  BEGIN
    DELETE FROM customers
    WHERE id = c_id;
  END delCustomer;
```

```
PROCEDURE listCustomer IS
  CURSOR c_customers is
    SELECT  name FROM customers;
  TYPE c_list is TABLE OF customers.name%type;
  name_list c_list := c_list();
  counter integer :=0;
  BEGIN
    FOR n IN c_customers LOOP
    counter := counter +1;
    name_list.extend;
    name_list(counter) := n.name;
    dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
    END LOOP;
  END listCustomer;

END c_package;
/
```

The above example makes use of the nested table.

When the above code is executed at the SQL prompt, it produces the following result –

Package body created.

Using The Package

The following program uses the methods declared and defined in the package c_package.

```
DECLARE
   code customers.id%type:= 5;
BEGIN
c_package.addcustomer(4, 'Raj', 27, 'Chandigar', 8500);
   c_package.addcustomer(5, 'Sakshi', 32, 'New Delhi', 9500);
   c_package.listcustomer;
   c_package.delcustomer(code);
   c_package.listcustomer;
END;
/
```

Customer(1): Ramesh
Customer(2): Khilan
Customer(3): kaushik
Customer(4): Chaitali
Customer(5): Hardik
Customer(6): Komal
Customer(7): Rajnish
Customer(8): Subham
Customer(1): Ramesh
Customer(2): Khilan
Customer(3): kaushik
Customer(4): Chaitali
Customer(5): Hardik
Customer(6): Komal
Customer(7): Rajnish

PL/SQL procedure successfully completed

```
SQL> create table customers(ID number,NAME varchar2(20),AGE number,ADDRESS varchar2(20),SALARY number(8,2));

Table created.

SQL> insert into customers values(1,'Nikita',25,'Delhi',8000);

1 row created.

SQL> insert into customers values(1,'Nikil',26,'Haryana',8500);

1 row created.

SQL> CREATE OR REPLACE PACKAGE c_package AS
  2    -- Adds a customer
  3    PROCEDURE addCustomer(c_id   customers.id%type,
  4    c_name  customers.name%type,
  5    c_age  customers.age%type,
  6    c_addr customers.address%type,
  7    c_sal  customers.salary%type);
  8
  9    -- Removes a customer
 10     PROCEDURE delCustomer(c_id  customers.id%TYPE);
 11     --Lists all customers
 12     PROCEDURE listCustomer;
 13
 14  END c_package;
 15  /

Package created.
```

```
SQL> CREATE OR REPLACE PACKAGE BODY c_package AS
  2    PROCEDURE addCustomer(c_id  customers.id%type,
  3      c_name customers.name%type,
  4      c_age  customers.age%type,
  5      c_addr  customers.address%type,
  6      c_sal   customers.salary%type)
  7    IS
  8    BEGIN
  9      INSERT INTO customers (id,name,age,address,salary)
 10         VALUES(c_id, c_name, c_age, c_addr, c_sal);
 11    END addCustomer;
 12
 13    PROCEDURE delCustomer(c_id   customers.id%type) IS
 14    BEGIN
 15      DELETE FROM customers
 16      WHERE id = c_id;
 17    END delCustomer;
 18  PROCEDURE listCustomer IS
 19    CURSOR c_customers is
 20      SELECT  name FROM customers;
 21    TYPE c_list is TABLE OF customers.name%type;
 22    name_list c_list := c_list();
 23    counter integer :=0;
 24    BEGIN
 25      FOR n IN c_customers LOOP
 26      counter := counter +1;
 27      name_list.extend;
 28      name_list(counter) := n.name;
 29      dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
 30      END LOOP;
 31    END listCustomer;
 32
 33  END c_package;
 34  /

Package body created.
```

```
SQL> DECLARE
  2    code customers.id%type:= 4;
  3  BEGIN
  4  c_package.addcustomer(3, 'Rajnish', 25, 'Chennai', 3500);
  5    c_package.addcustomer(4, 'Subham', 32, 'Delhi', 7500);
  6    c_package.listcustomer;
  7    c_package.delcustomer(code);
  8    c_package.listcustomer;
  9  END;
 10  /
Customer(1)Nikita
Customer(2)Nikil
Customer(3)Rajnish
Customer(4)Subham
Customer(1)Nikita
Customer(2)Nikil
Customer(3)Rajnish

PL/SQL procedure successfully completed.

SQL>
```

Run SQL Command Line

```
SQL> create table customers(ID number,NAME varchar2(20),AGE number,ADDRESS varchar2(20),SALARY number(8,2));

Table created.

SQL> insert into customers values(1,'Nikita',25,'Delhi',8000);

1 row created.

SQL> insert into customers values(1,'Nikil',26,'Haryana',8500);

1 row created.

SQL> CREATE OR REPLACE PACKAGE c_package AS
  2     -- Adds a customer
  3     PROCEDURE addCustomer(c_id   customers.id%type,
  4     c_name  customers.name%type,
  5     c_age  customers.age%type,
  6     c_addr customers.address%type,
  7     c_sal  customers.salary%type);
  8
  9     -- Removes a customer
 10     PROCEDURE delCustomer(c_id  customers.id%TYPE);
 11     --Lists all customers
 12     PROCEDURE listCustomer;
 13
 14  END c_package;
 15  /

Package created.

SQL> CREATE OR REPLACE PACKAGE BODY c_package AS
  2     PROCEDURE addCustomer(c_id  customers.id%type,
  3        c_name customers.name%type,
  4        c_age  customers.age%type,
  5        c_addr  customers.address%type,
  6        c_sal   customers.salary%type)
  7     IS
  8     BEGIN
  9        INSERT INTO customers (id,name,age,address,salary)
 10           VALUES(c_id, c_name, c_age, c_addr, c_sal);
 11     END addCustomer;
 12
 13     PROCEDURE delCustomer(c_id   customers.id%type) IS
 14     BEGIN
```

Run SQL Command Line

```
 13
 14  END c_package;
 15  /

Package created.

SQL> CREATE OR REPLACE PACKAGE BODY c_package AS
  2     PROCEDURE addCustomer(c_id  customers.id%type,
  3        c_name customers.name%type,
  4        c_age  customers.age%type,
  5        c_addr  customers.address%type,
  6        c_sal   customers.salary%type)
  7     IS
  8     BEGIN
  9        INSERT INTO customers (id,name,age,address,salary)
 10           VALUES(c_id, c_name, c_age, c_addr, c_sal);
 11     END addCustomer;
 12
 13     PROCEDURE delCustomer(c_id   customers.id%type) IS
 14     BEGIN
 15        DELETE FROM customers
 16        WHERE id = c_id;
 17     END delCustomer;
 18  PROCEDURE listCustomer IS
 19     CURSOR c_customers is
 20        SELECT  name FROM customers;
 21     TYPE c_list is TABLE OF customers.name%type;
 22     name_list c_list := c_list();
 23     counter integer :=0;
 24     BEGIN
 25        FOR n IN c_customers LOOP
 26        counter := counter +1;
 27        name_list.extend;
 28        name_list(counter) := n.name;
 29        dbms_output.put_line('Customer(' ||counter|| ')'||name_list(counter));
 30        END LOOP;
 31     END listCustomer;
 32
 33  END c_package;
 34  /

Package body created.

SQL> DECLARE
```

Run SQL Command Line

```
 34  /

Package body created.

SQL> DECLARE
  2     code customers.id%type:= 4;
  3  BEGIN
  4  c_package.addcustomer(3, 'Rajnish', 25, 'Chennai', 3500);
  5     c_package.addcustomer(4, 'Subham', 32, 'Delhi', 7500);
  6     c_package.listcustomer;
  7     c_package.delcustomer(code);
  8     c_package.listcustomer;
  9  END;
 10  /
Customer(1)Nikita
Customer(2)Nikil
Customer(3)Rajnish
Customer(4)Subham
Customer(1)Nikita
Customer(2)Nikil
Customer(3)Rajnish

PL/SQL procedure successfully completed.

SQL>
```


Thank You