



# Database Programming

## 6.Functions

## 6.Functions

A function that is stored in the database is much like a procedure, in that it is a named PL/SQL block that can take parameters and be invoked.

### Creating Functions

Functions are another type of stored code and are very similar to procedures. The significant difference between the two is that **a function is a PL/SQL block that returns a single value**. Functions can accept one, many, or no parameters, but they must have a return clause in their execution section. The data type of the return value must be declared in the header of the function. A function is not a stand-alone executable in the same way that a procedure is; that is, a function must always be used in some context.

# 6.Functions

## Creating Stored Functions

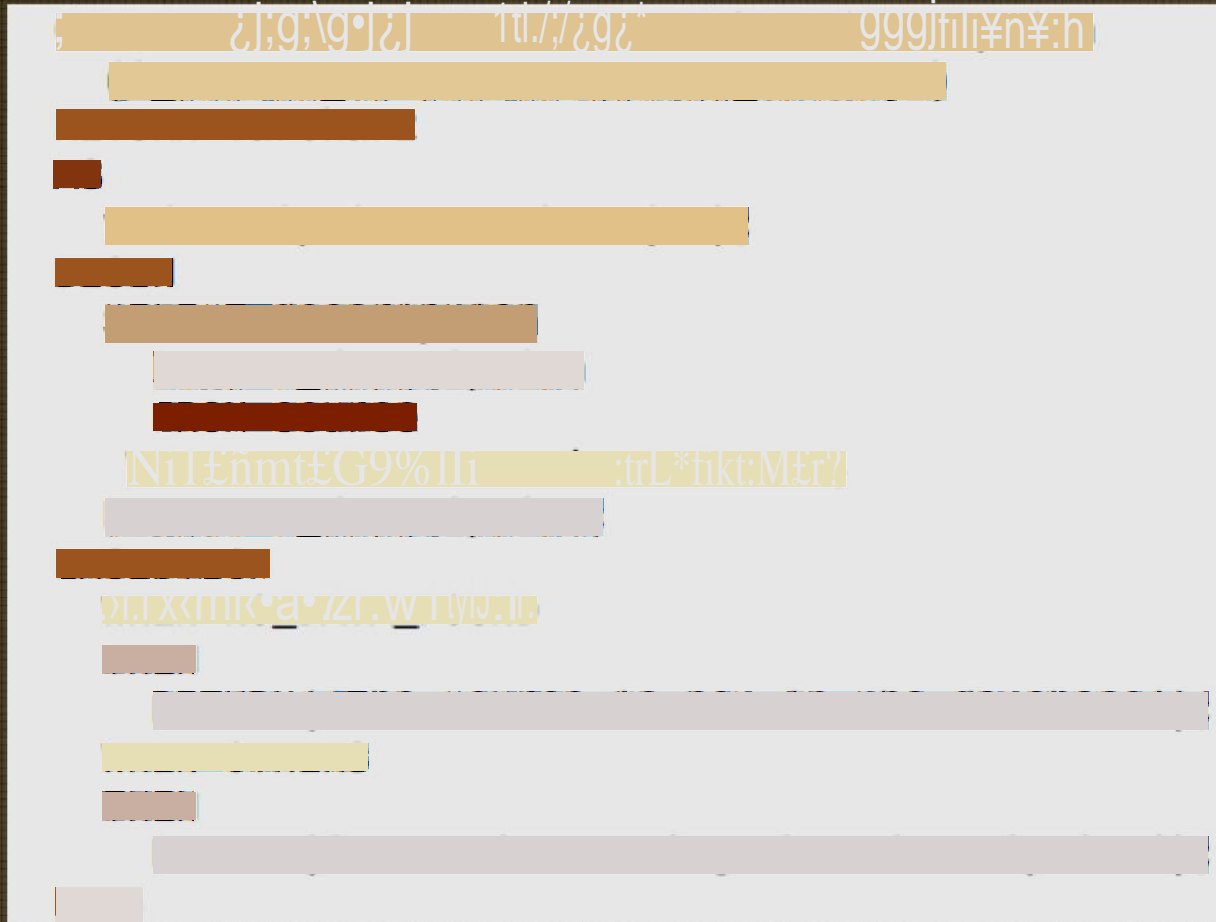
The syntax for creating a function is as follows:

```
CREATE [OR REPLACE] FUNCTION function_name  
  (parameter list)  
  RETURN datatype  
IS  
BEGIN  
  <body>  
  RETURN (return_value);  
END;
```

The function does not necessarily have any parameters, but it must have a RETURN value declared in the header, and it must return values for all of the possible execution streams

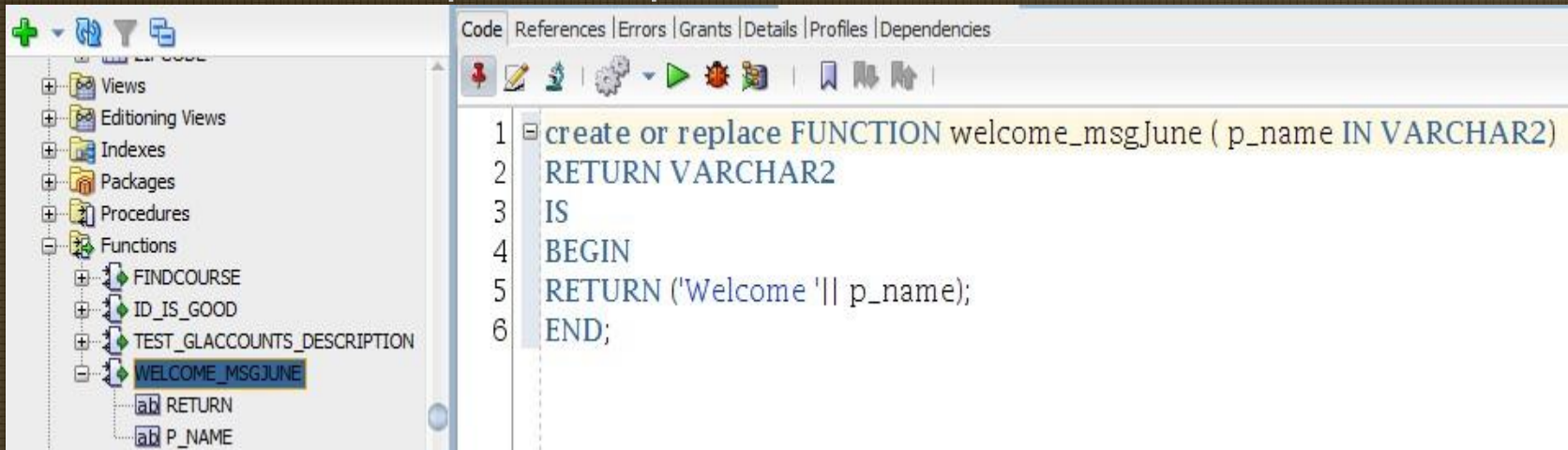


## 6.Functions

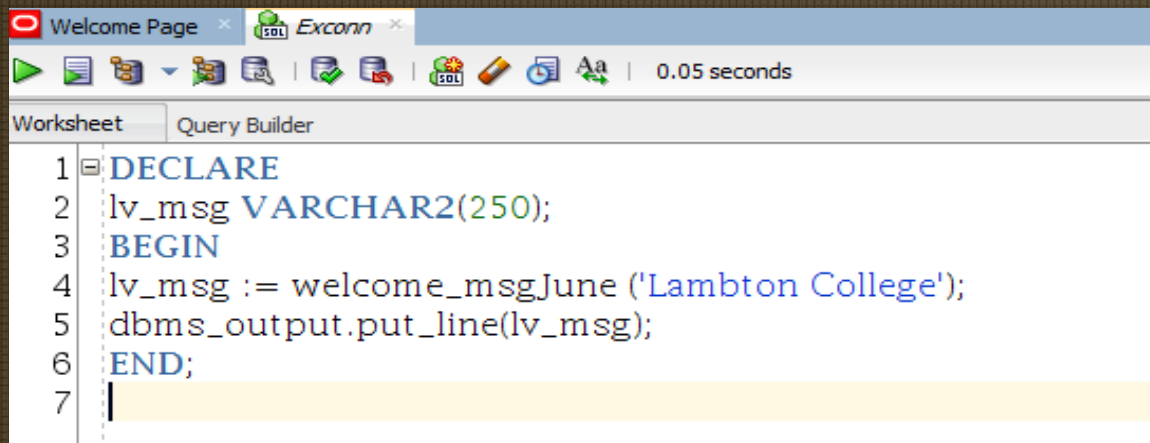


## 6.Functions

Lets consider a simple example

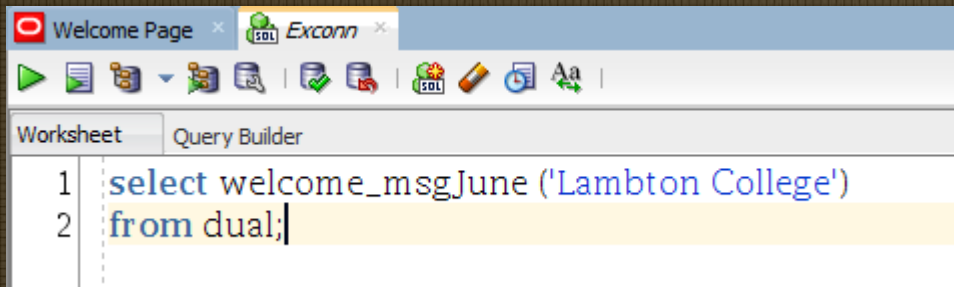


Now execute the code using following method (Method 1)



## 6.Functions

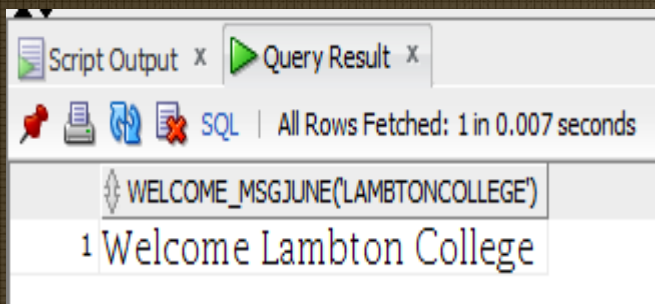
Now execute the code using second method (Method 2)



```
1 select welcome_msgJune ('Lambton College')
2 from dual;
```

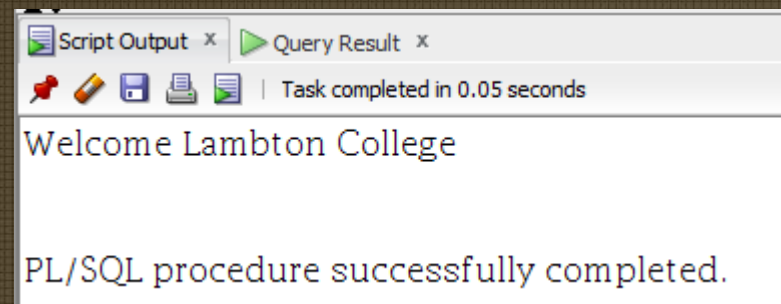
Following is the output :

Second Method :



```
WELCOME_MSGJUNE('LAMBTONCOLLEGE')
1 Welcome Lambton College
```

First Method



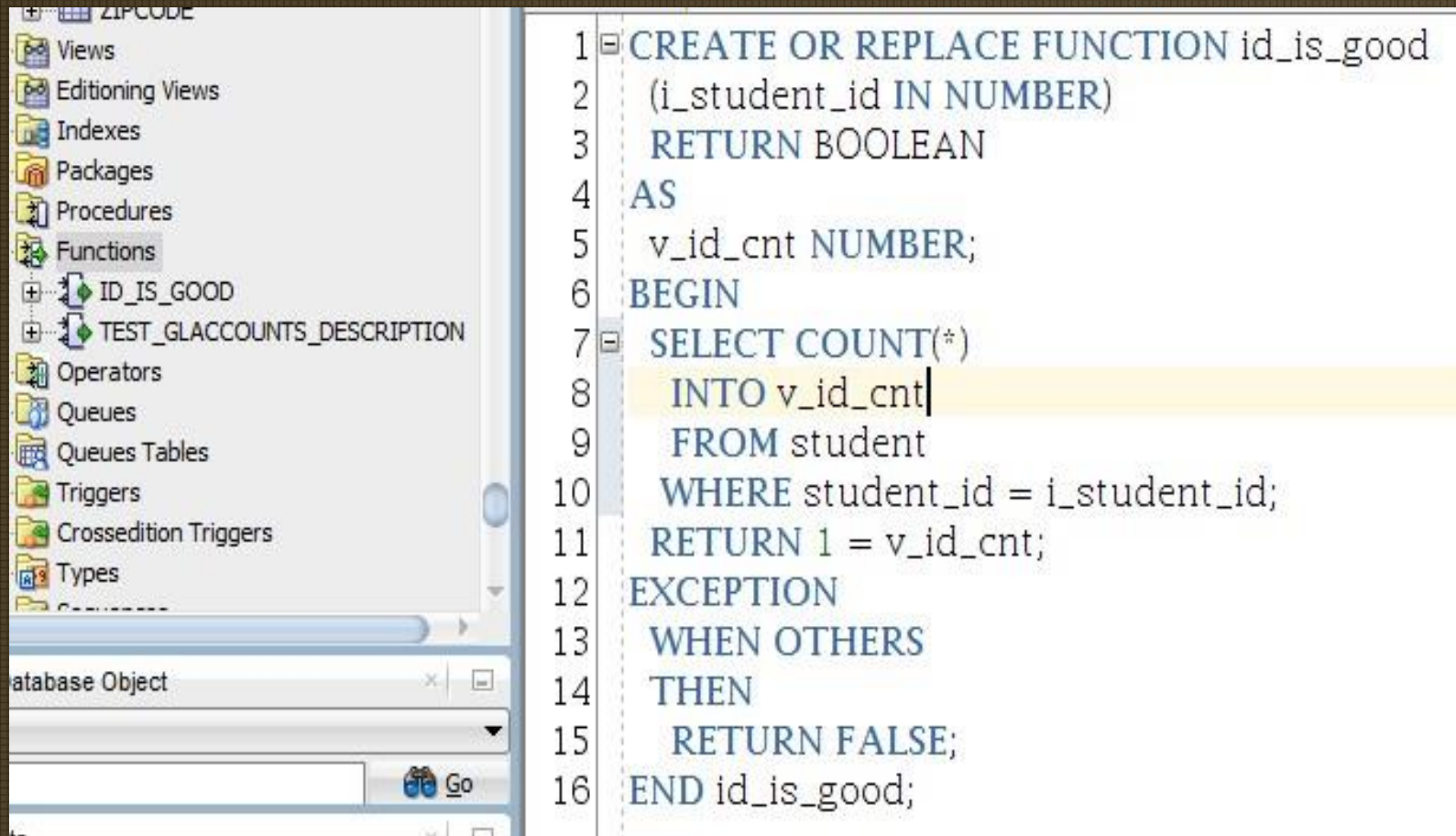
```
Welcome Lambton College

PL/SQL procedure successfully completed.
```



## 6.Functions

Lets consider another example makes use of the function.



The screenshot shows a database IDE interface. On the left, a tree view displays database objects: Views, Editing Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, Crossedition Triggers, and Types. Under the 'Functions' folder, two functions are listed: 'ID\_IS\_GOOD' and 'TEST\_GLACCOUNTS\_DESCRIPTION'. The 'ID\_IS\_GOOD' function is selected. On the right, the SQL editor displays the definition of the 'ID\_IS\_GOOD' function. The code is as follows:

```
1 CREATE OR REPLACE FUNCTION id_is_good
2   (i_student_id IN NUMBER)
3   RETURN BOOLEAN
4 AS
5   v_id_cnt NUMBER;
6 BEGIN
7   SELECT COUNT(*)
8     INTO v_id_cnt
9     FROM student
10    WHERE student_id = i_student_id;
11   RETURN 1 = v_id_cnt;
12 EXCEPTION
13   WHEN OTHERS
14   THEN
15     RETURN FALSE;
16 END id_is_good;
```

Explain line number 11 in class

## 6.Functions

The screenshot displays the Oracle SQL Developer environment. The main window is titled 'Exconn' and shows a PL/SQL script in the 'Query Builder' tab. The script is as follows:

```
1 DECLARE
2   V_id number;
3 BEGIN
4   V_id := &id;
5 IF id_is_good(v_id)
6 THEN
7   DBMS_OUTPUT.PUT_LINE
8 ('Student ID: '||v_id||' is a valid. ');
9 ELSE
10  DBMS_OUTPUT.PUT_LINE
11 ('Student ID: '||v_id||' is not valid. ');
12 END IF;
13 END;
```

An 'Enter Substitution Variable' dialog box is open, prompting the user to 'Enter value for id:'. The input field contains the value '102'. The dialog has 'OK' and 'Cancel' buttons.

At the bottom of the window, the 'Script Output' tab is visible, showing the execution result:

```
END;
Student ID: 102 is a valid.
```



## 6.Functions

Lets consider one more example of functions.

The screenshot displays the Oracle SQL Developer environment. On the left, the 'Database Object' tree shows the 'FUNCTIONS' folder expanded, with 'SHOW\_DESCRIPTION' selected. Below this, the 'Reports' section is visible. The main 'Code' editor on the right contains the following SQL code:

```
1 create or replace FUNCTION show_description
2 (i_course_code courses.code%TYPE)
3 RETURN varchar2
4 AS
5 --pragma UDF;
6 v_description varchar2(50);
7 BEGIN
8 SELECT description
9 INTO v_description
10 FROM courses
11 WHERE code = i_course_code;
12 RETURN v_description;
13 EXCEPTION
14 WHEN NO_DATA_FOUND
15 THEN
16 RETURN('The Course is not in the database');
17 WHEN OTHERS
18 THEN
19 RETURN('Error in running show_description');
20 END;
```

## 6.Functions

Following is the output :

