Name: - Suyash Navanath Shinde Roll No: - D2223034
Course: - Database Management System Lab Batch: - B
Division: - TY CSF 2

Experiment No. 6

Aim :- To study PL/SQL: Procedures and Functions.

Title :-

1) Write a function to square the number taken from the user.
2) Write a procedure to display the records from Manufacturing industry / Hospital/ Company table

Software Required - Oracle Server, SQLite

Theory :-

The PL/SQL stored procedure or simply a procedure is a PL/SQL block which performs one or more specific tasks. It is just like procedures in other programming languages.

The procedure contains a header and a body.

- Header: The header contains the name of the procedure and the parameters or variables passed to the procedure.
- Body: The body contains a declaration section, execution section and exception section similar to a general PL/SQL block.

- How to pass parameters in procedure:

When you want to create a procedure or function, you have to define parameters .There is three ways to pass parameters in procedure:

1. IN parameters: The IN parameter can be referenced by the procedure or function. The value of the parameter cannot be overwritten by the procedure or the function.
2. OUT parameters: The OUT parameter cannot be referenced by the procedure or function, but the value of the parameter can be overwritten by the procedure or function.
3. INOUT parameters: The INOUT parameter can be referenced by the procedure or function and the value of the parameter can be overwritten by the procedure or function.

Syntax for creating procedure:

```
1. CREATE [OR REPLACE] PROCEDURE procedure_name
2.    [ (parameter [,parameter]) ]
3. IS
4.    [declaration_section]
```

```
BEGIN
    executable_section
[EXCEPTION
    exception_section]
END [procedure_name];
```

- Below are the characteristics of Procedure subprogram unit in PL/SQL:

1) Procedures are standalone blocks of a program that can be stored in the database.

2) Call to these PL/SQL procedures can be made by referring to their name, to execute the PL/SQL statements.

3) It is mainly used to execute a process in PL/SQL.

4) It can have nested blocks, or it can be defined and nested inside the other blocks or packages.

5) It contains declaration part (optional), execution part, exception handling part (optional).

6) The values can be passed into Oracle procedure or fetched from the procedure through parameters.

7) These parameters should be included in the calling statement.

8) A Procedure in SQL can have a RETURN statement to return the control to the calling block, but it cannot return any values through the RETURN statement.

9) Procedures cannot be called directly from SELECT statements. They can be called from another block or through EXEC keyword.

Example1: Creating Procedure and calling it using EXEC

In this example, we are going to create an Oracle procedure that takes the name as input and prints the welcome message as output. We are going to use EXEC command to call procedure.

CODE:

- **PL/SQL Function** :-

Functions is a standalone PL/SQL subprogram. Like PL/SQL procedure, functions have a unique name by which it can be referred. These are stored as PL/SQL database objects.

Below are some of the characteristics of functions.

1)Functions are a standalone block that is mainly used for calculation purpose.

2)Function use RETURN keyword to return the value, and the datatype of this is defined at the time of creation.

3)It can have nested blocks, or it can be defined and nested inside the other blocks or packages.

4)It contains the declaration part (optional), execution part, exception handling part (optional).

5) The values can be passed into the function or fetched from the procedure through the parameters.

6)These parameters should be included in the calling statement. A PLSQL function can also return the value through OUT parameters other than using RETURN.

Syntax

```
CREATE OR REPLACE FUNCTION
<procedure_name>
(
<parameterl IN/OUT <datatype>
)
RETURN <datatype>
[ IS | AS ]
<declaration_part>
BEGIN
<execution part>
EXCEPTION
<exception handling part>
END;
```

A) Write a function to square the number taken from user

QUERY: -

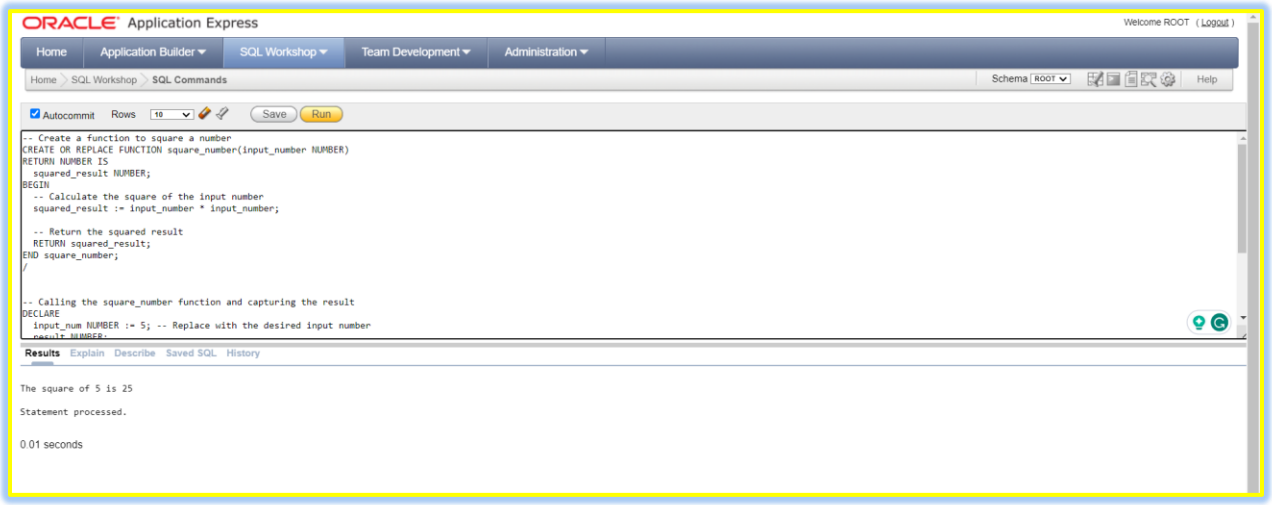
```
-- Create a function to square a number
CREATE OR REPLACE FUNCTION square_number(input_number NUMBER)
RETURN NUMBER IS
  squared_result NUMBER;
BEGIN
  -- Calculate the square of the input number
  squared_result := input_number * input_number;

  -- Return the squared result
  RETURN squared_result;
END square_number;
/


-- Calling the square_number function and capturing the result
DECLARE
  input_num NUMBER := 5; -- Replace with the desired input number
  result NUMBER;
BEGIN
  -- Call the square_number function and store the result in the 'result'
variable
  result := square_number(input_num);

  -- Display the result
  DBMS_OUTPUT.PUT_LINE('The square of ' || input_num || ' is ' || result);
END;
/
```

OUTPUT: -



B) Write a procedure to display the records from the Manufacturing industry / Hospital/ Company table.

QUERY: -

```
-- Create a PL/SQL procedure to display records from the Company table
CREATE OR REPLACE PROCEDURE display_company_records IS
BEGIN
  -- Declare a cursor to retrieve records from the Company table
  CURSOR company_cursor IS
    SELECT * FROM Company;

  -- Declare variables to store column values
  company_id Company.company_id%TYPE;
  company_name Company.company_name%TYPE;
  company_address Company.company_address%TYPE;
  -- Add more variables for additional columns as needed

  -- Open the cursor
  OPEN company_cursor;

  -- Loop through the cursor results and display each record
  LOOP
    FETCH company_cursor INTO company_id, company_name, company_address;
    EXIT WHEN company_cursor%NOTFOUND;

    -- Display the record (you can modify the output format as needed)
    DBMS_OUTPUT.PUT_LINE('Company ID: ' || company_id);
    DBMS_OUTPUT.PUT_LINE('Company Name: ' || company_name);
```

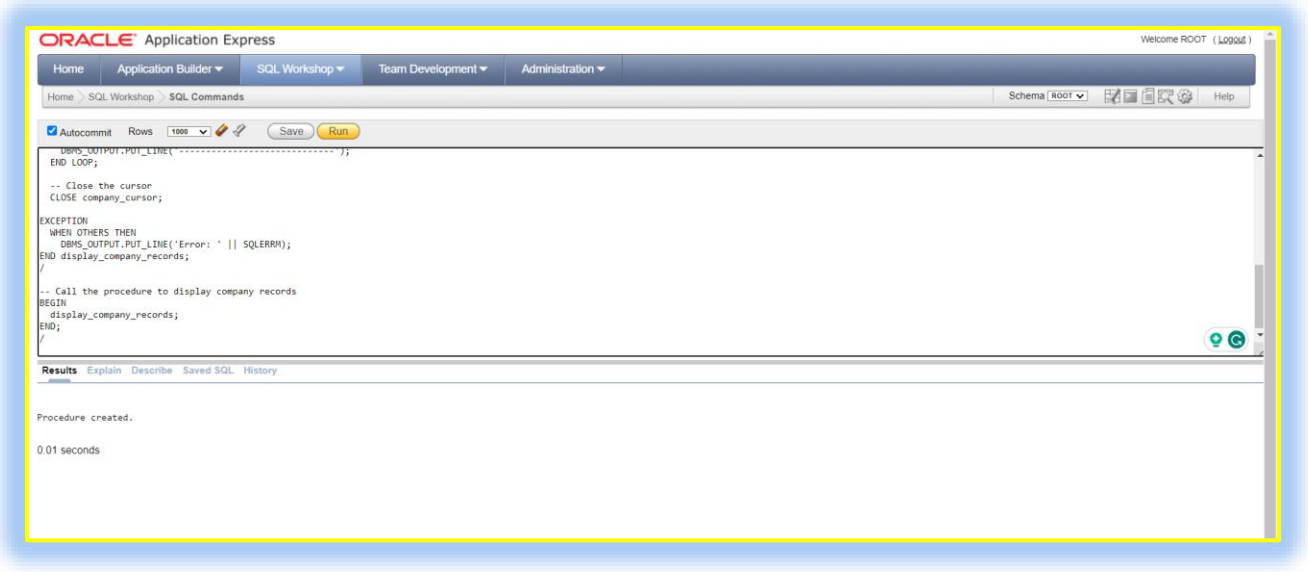
```
    DBMS_OUTPUT.PUT_LINE('Company Address: ' || company_address);
    -- Add more lines to display additional columns if needed
    DBMS_OUTPUT.PUT_LINE('-----------------------------');
  END LOOP;

  -- Close the cursor
  CLOSE company_cursor;

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
END display_company_records;
/
```

```
-- Call the procedure to display company records
BEGIN
  display_company_records;
END;
/
```

OUTPUT: -



Exercise –

1)Write a PL/SQL program using WHILE loop for calculating the average of the numbers entered by the user. Stop the entry of numbers whenever the user enters the number 0.

2) Write a PL/SQL code to find whether a given string is palindrome or not.

3) Write a PL/SQL program to find the sum of digits of a number.

4) Write a procedure to display the records from the Manufacturing industry table.

5) Write a procedure to display the records from the Hospital table.

FAQ:-

1)Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees.

2) Write a PL/SQL program to display the names of all countries.

3) What is the Difference Between Pl SQL and SQL?

4) What is an Alias in SQL Statements?

5) What is a Dual Table?

6) What is Invalid_number, Value_error?

7)Explain Different Methods to Trace the Pl/SQL Code?