Lab 4. PL/SQL Control Structures, Procedures and Functions

Submission:

- If you decide to skip the lab, make sure you submit the **check-off questions** (highlighted with green background) in the text box of the check-off assignment item for this Lab before **Wednesday, noon to Brightspace**.
- **All students are expected to submit your answers to Table 1 and Table 2, or Table 1 and Table 3 in a text document with the name <lastname>_<firstname>_lab4.sql or .txt by the due date to Gradescope.**
- Please include both your code and the results in the **.txt** or **.sql** documents for full credits. For detailed requirements, please refer to the "Lab and Homework Submission Guideline."

Objectives:

- Practice implementing PL/SQL Control Structures
- Practice implementing PL/SQL Procedures and Functions

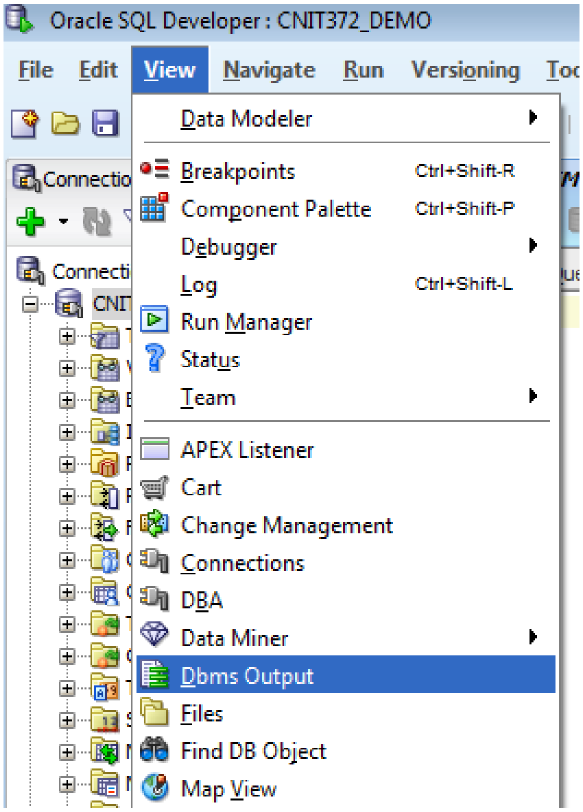
Notes:

- The first table of questions is about control structure. Every student is expected to complete this section of the lab (Table 1).
- The second and third tables are about procedures and functions. We would like to give an opportunity (Table 3) to students who are familiar with PL/SQL syntax and programming logic to have a more challenging task.
- Table 2 provides step-by-step instructions. Table 3 is more challenging and requires you to make a lot of choices on your own.
- **To recap, your submission should include answers to Table 1 + Table 2, or Table 1 + Table 3.**

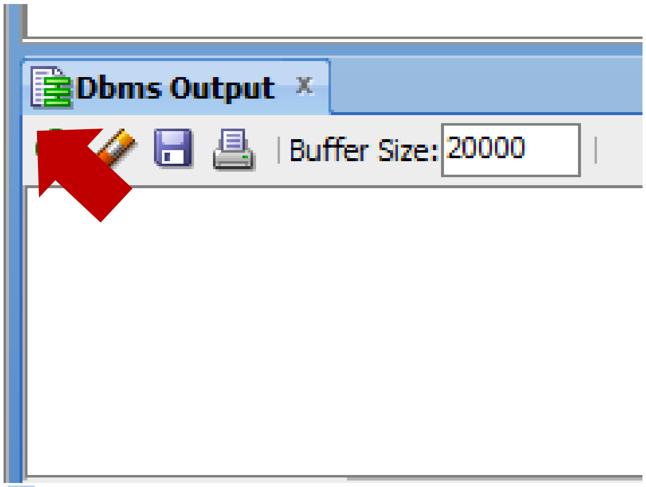
Part A was developed by Ronald Erdei

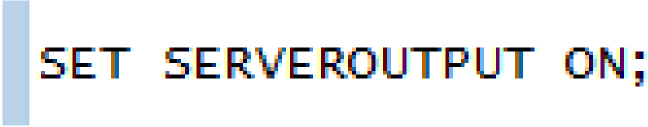
Preparation: Using the Oracle SQL Developer DBMS Output Pane

1. Turn on the DBMS Output Pane in SQL Developer



2. Set the pane to display the DBMS Output from your account by clicking the green plus sign (+) and selecting the appropriate connection.

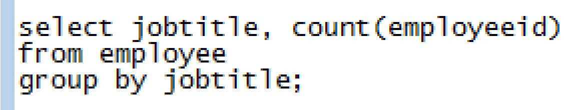
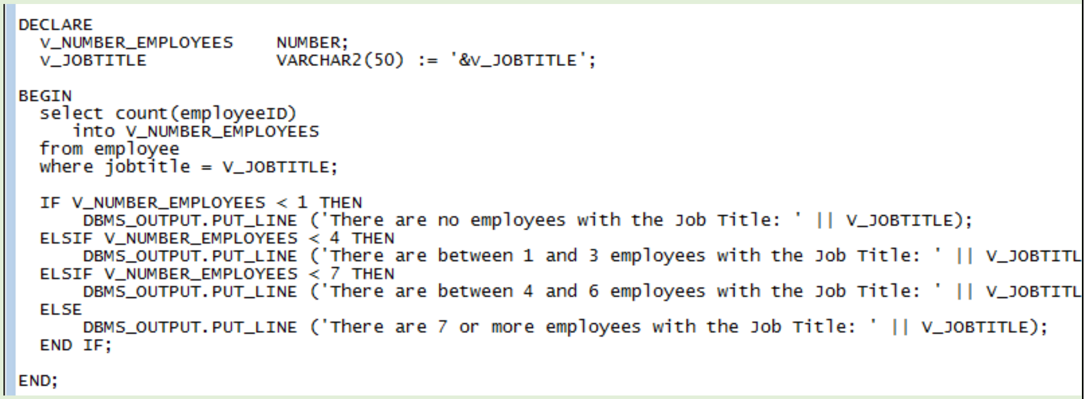




```
SET SERVEROUTPUT ON;
```

Part A was developed by Ronald Erdei

Table 1: Control Structures (6.1 pts)

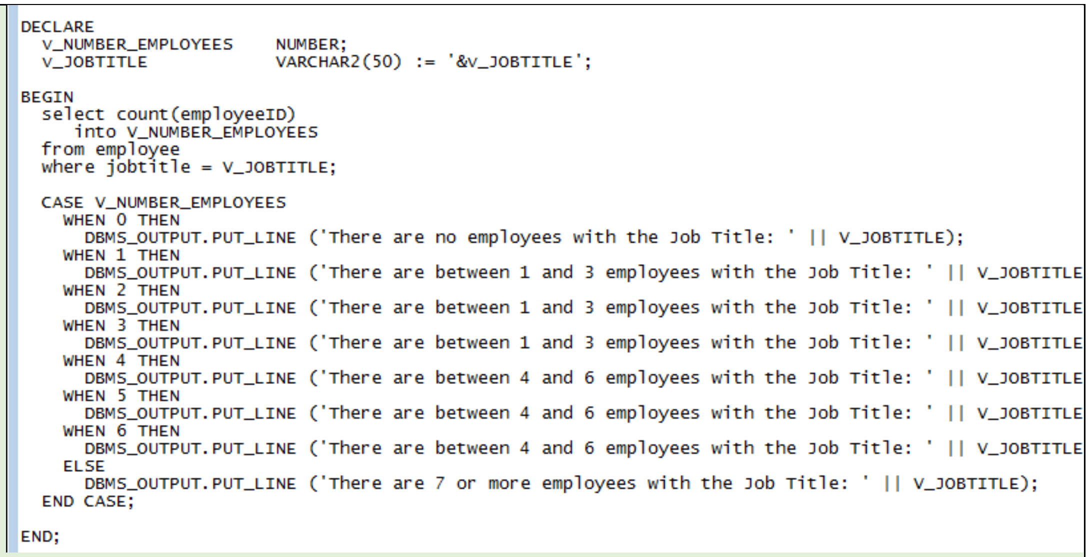
<table>
<tr><td colspan="3"><b>Conditional Selection Statements</b></td></tr>
<tr><td colspan="3"><b>Basic IF … THEN conditional structure</b></td></tr>
<tr><td><b>1a</b></td><td>0.2</td><td>Run the following SQL query:
<pre>
select jobtitle, count(employeeid)
from employee
group by jobtitle;
</pre>What are the results of this query?</td></tr>
<tr><td><b>1b</b></td><td>0.2</td><td>Run the following PL/SQL code:
<pre>
DECLARE
  V_NUMBER_EMPLOYEES   NUMBER;
  V_JOBTITLE   VARCHAR2(50) := '&v_JOBTITLE';

BEGIN
  select count(employeeID)
     into V_NUMBER_EMPLOYEES
  from employee
  where jobtitle = V_JOBTITLE;

  IF V_NUMBER_EMPLOYEES < 1 THEN
      DBMS_OUTPUT.PUT_LINE ('There are no employees with the Job Title: ' || V_JOBTITLE);
  ELSIF V_NUMBER_EMPLOYEES < 4 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 1 and 3 employees with the Job Title: ' || V_JOBTITL
  ELSIF V_NUMBER_EMPLOYEES < 7 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 4 and 6 employees with the Job Title: ' || V_JOBTITL
  ELSE
      DBMS_OUTPUT.PUT_LINE ('There are 7 or more employees with the Job Title: ' || V_JOBTITLE);
  END IF;

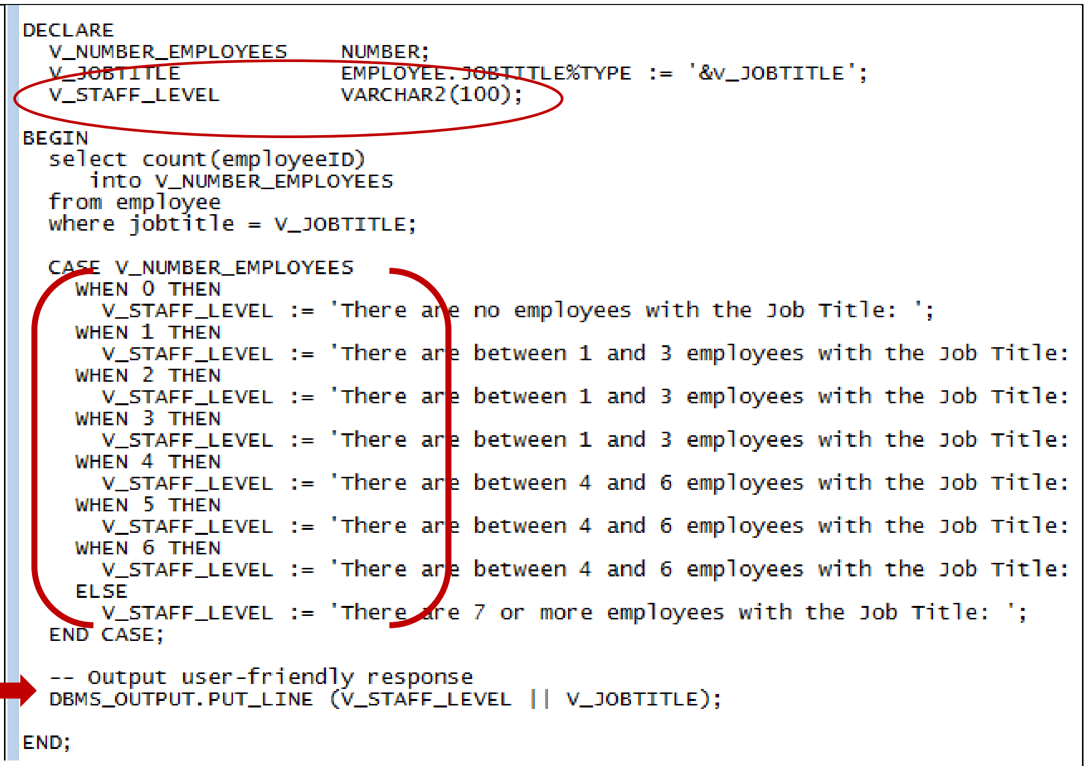
END;
</pre>When prompted for Job Title, provide the following value: CIO (this is case sensitive)
What is the output (e.g., what is printed to the screen)?</td></tr>
<tr><td><b>1c</b></td><td>0.2</td><td>Again, run the PL/SQL code from 1b. When prompted for Job Title, provide the following value: Accountant (this is case sensitive)
What is the output (e.g., what is printed to the screen)?</td></tr>
<tr><td><b>1d</b></td><td>0.2</td><td>Again, run the PL/SQL code from 1b. When prompted for Job Title, provide the following value: Engineer (this is case sensitive)
What is the output (e.g., what is printed to the screen)?</td></tr>
<tr><td><b>1e</b></td><td>0.2</td><td>Again, run the PL/SQL code from 1b. When prompted for Job Title, provide the following value: Assembly (this is case sensitive)
What is the output (e.g., what is printed to the screen)?</td></tr>
<tr><td></td><td colspan="2"><b>Basic CASE conditional structure</b></td></tr>
<tr><td><b>2a</b></td><td>0.2</td><td>Run the following PL/SQL code:</td></tr>
</table>

Part A was developed by Ronald Erdei



```
DECLARE
  V_NUMBER_EMPLOYEES     NUMBER;
  V_JOBTITLE             VARCHAR2(50) := '&V_JOBTITLE';

BEGIN
  select count(employeeID)
     into V_NUMBER_EMPLOYEES
  from employee
  where jobtitle = V_JOBTITLE;

  CASE V_NUMBER_EMPLOYEES
    WHEN 0 THEN
      DBMS_OUTPUT.PUT_LINE ('There are no employees with the Job Title: ' || V_JOBTITLE);
    WHEN 1 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 1 and 3 employees with the Job Title: ' || V_JOBTITLE
    WHEN 2 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 1 and 3 employees with the Job Title: ' || V_JOBTITLE
    WHEN 3 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 1 and 3 employees with the Job Title: ' || V_JOBTITLE
    WHEN 4 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 4 and 6 employees with the Job Title: ' || V_JOBTITLE
    WHEN 5 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 4 and 6 employees with the Job Title: ' || V_JOBTITLE
    WHEN 6 THEN
      DBMS_OUTPUT.PUT_LINE ('There are between 4 and 6 employees with the Job Title: ' || V_JOBTITLE
    ELSE
      DBMS_OUTPUT.PUT_LINE ('There are 7 or more employees with the Job Title: ' || V_JOBTITLE);
  END CASE;

END;
```

		When prompted for Job Title, provide the following value: CIO. What is the output (e.g., what is printed to the screen)?
2b	0.2	Again, run the PL/SQL code from 2a. When prompted for Job Title, provide the following value: Accountant. What is the output (e.g., what is printed to the screen)?
2c	0.2	Again, run the PL/SQL code from 2a. When prompted for Job Title, provide the following value: Engineer. What is the output (e.g., what is printed to the screen)?
2d	0.2	Again, run the PL/SQL code from 2a. When prompted for Job Title, provide the following value: Assembly. What is the output (e.g., what is printed to the screen)?
Refining the Code Refine your code to be more programmatically efficient in its structure through the use of variables in your **CASE** conditional statement.		
3	1.5	Modify the PL/SQL code from 2a, such that a. the data type of the variable **V_JOBTITLE** has the same data type as the column **jobtitle**, no matter how the schema changes. b. you can use only a single call to the DBMS_OUTPUT.PUT_LINE procedure. Use the local variable V_STAFF_LEVEL to facilitate this, as shown below. Then run the code.



```
DECLARE
  V_NUMBER_EMPLOYEES     NUMBER;
  V_JOBTITLE             EMPLOYEE.JOBTITLE%TYPE := '&V_JOBTITLE';
  V_STAFF_LEVEL          VARCHAR2(100);

BEGIN
  select count(employeeID)
     into V_NUMBER_EMPLOYEES
  from employee
  where jobtitle = V_JOBTITLE;

  CASE V_NUMBER_EMPLOYEES
    WHEN 0 THEN
      V_STAFF_LEVEL := 'There are no employees with the Job Title: ';
    WHEN 1 THEN
      V_STAFF_LEVEL := 'There are between 1 and 3 employees with the Job Title:
    WHEN 2 THEN
      V_STAFF_LEVEL := 'There are between 1 and 3 employees with the Job Title:
    WHEN 3 THEN
      V_STAFF_LEVEL := 'There are between 1 and 3 employees with the Job Title:
    WHEN 4 THEN
      V_STAFF_LEVEL := 'There are between 4 and 6 employees with the Job Title:
    WHEN 5 THEN
      V_STAFF_LEVEL := 'There are between 4 and 6 employees with the Job Title:
    WHEN 6 THEN
      V_STAFF_LEVEL := 'There are between 4 and 6 employees with the Job Title:
    ELSE
      V_STAFF_LEVEL := 'There are 7 or more employees with the Job Title: ';
  END CASE;

  -- output user-friendly response
  DBMS_OUTPUT.PUT_LINE (V_STAFF_LEVEL || V_JOBTITLE);

END;
```

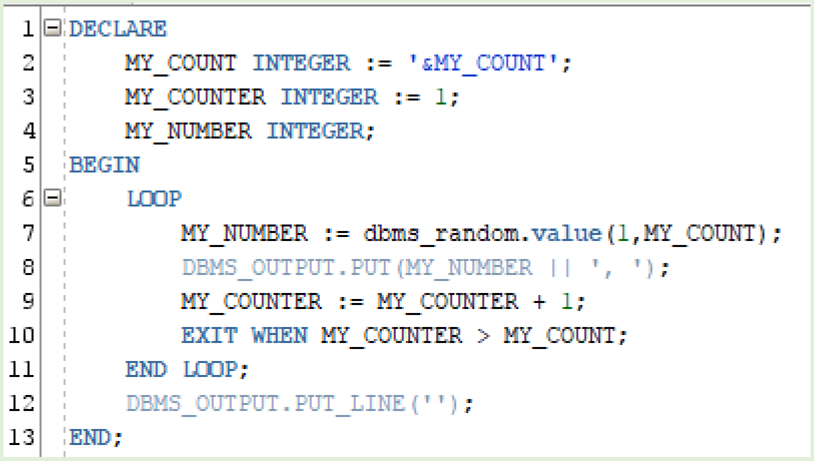
When prompted for Job Title, provide the following value: Engineer.

What is the output (e.g., what is printed to the screen)?

Loop Constructs

Creating a "Simple" Loop in PL/SQL

4a 0.2 Using the following code, create a simple loop.



```
DECLARE
    MY_COUNT INTEGER := '&MY_COUNT';
    MY_COUNTER INTEGER := 1;
    MY_NUMBER INTEGER;
BEGIN
    LOOP
        MY_NUMBER := dbms_random.value(1,MY_COUNT);
        DBMS_OUTPUT.PUT(MY_NUMBER || ', ');
        MY_COUNTER := MY_COUNTER + 1;
        EXIT WHEN MY_COUNTER > MY_COUNT;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('');
END;
```

When prompted for MY_COUNT, provide the value **5**.

What is the output (e.g., what is printed to the screen)?

4b 0.2 Explain what the simple loop "does".

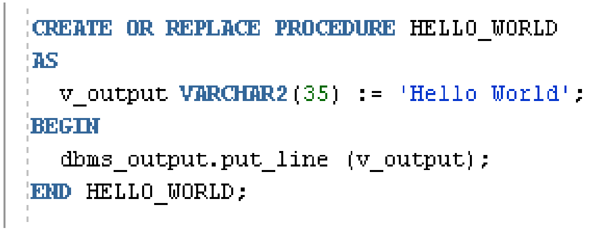
Creating a WHILE Loop in PL/SQL

4a 1 Rewrite the above simple loop with WHILE Loop.

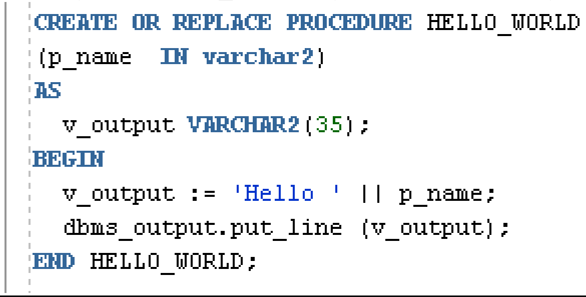
4b	0.2	Verify your WHILE LOOP by running it. Include the resulting output in your submission.
		Creating a FOR Loop in PL/SQL
5a	1	Rewrite the simple loop in 4a with FOR loop.
5b	0.2	Verify your FOR LOOP by running it. Include the resulting output in your submission.

Table 2: Procedures and Functions (18.9 pts)

Creating a (parameter-less) Procedure 0.5 * 4 = 2 pts	

1a	Run the following PL/SQL code:
	
	`CREATE OR REPLACE PROCEDURE HELLO_WORLD`
	`AS`
	`  v_output VARCHAR2(35) := 'Hello World';`
	`BEGIN`
	`  dbms_output.put_line (v_output);`
	`END HELLO_WORLD;`
	What is the output (e.g., what is printed to the screen)? To which SQL Developer pane is it printed?

1b	What is the SQL code necessary to execute the HELLO_WORLD procedure?
1c	Execute the procedure with your code. What is printed to the DBMS Output pane?
1d	What is the difference between anonymous blocks and named blocks? Here is a helpful reference: https://blogs.oracle.com/oraclemagazine/building-with-blocks
Default Coding Standards (e.g., formatting) for PL/SQL code The following method can be used to apply Oracle recommended capitalization to PL/SQL code. If desired, custom templates for formatting can be created. (0.4 pt)	
2	Highlight the code from 1a. Right click and then select: **FORMAT** What is the result? Include the formatted code in your answer file.
Creating a Procedure that accepts a single IN parameter (0.5 * 5 = 2.5 pts)	

3	Modify the HELLO_WORLD procedure to accept and use a single IN parameter as shown below.
	
	`CREATE OR REPLACE PROCEDURE HELLO_WORLD`
	`(p_name  IN varchar2)`
	`AS`
	`  v_output VARCHAR2(35);`
	`BEGIN`
	`  v_output := 'Hello ' || p_name;`
	`  dbms_output.put_line (v_output);`
	`END HELLO_WORLD;`

3a	If you wanted the output to read 'Hello World', what is the SQL code necessary to execute the HELLO_WORLD procedure? Execute the procedure with your code. What is printed to the DBMS Output pane?
3b	Execute the HELLO_WORLD procedure, passing it the following parameter: **Mark**. What is printed to the DBMS Output pane?
3c	Execute the HELLO_WORLD procedure, passing it the following parameter: **World procedure. I have so much fun coding in SQL and PLSQL.** What is printed to the Script Output pane?
3d	Modify the HELLO_WORLD procedure so that it will work with the parameter shown in question 3c. Include the revised definition for the HELLO_WORLD procedure in your submission.
3e	Execute the HELLO_WORLD procedure, passing it the following parameter:

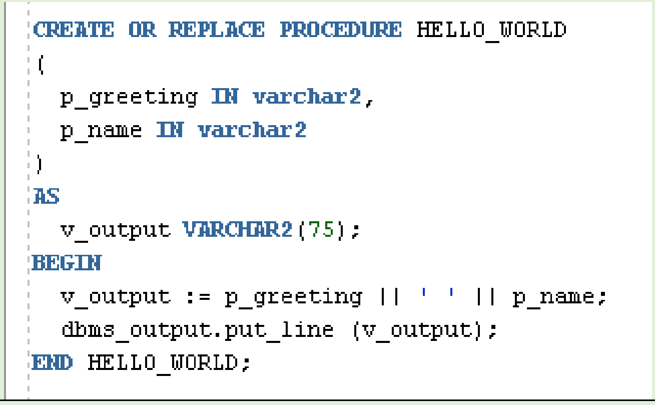
World procedure. I have so much fun coding in SQL and PLSQL.

What is printed to the DBMS Output pane?

Creating a Procedure that accepts multiple IN parameters

0.5 * 4 = 2 pts

4 Modify the HELLO_WORLD procedure to accept and use two IN parameters as shown below.

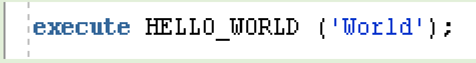


```
CREATE OR REPLACE PROCEDURE HELLO_WORLD
(
  p_greeting IN varchar2,
  p_name IN varchar2
)
AS
  v_output VARCHAR2(75);
BEGIN
  v_output := p_greeting || ' ' || p_name;
  dbms_output.put_line (v_output);
END HELLO_WORLD;
```

4a If you wanted the output to read ‘Hello World’, what is the SQL code necessary to execute the HELLO_WORLD procedure?

Execute the procedure with your code. What is printed to the DBMS Output pane?

4b Run the following line of SQL code:



```
execute HELLO_WORLD ('World');
```

What is printed to the Script Output pane? You should get an error. What type of error is this?

4c If you wanted the output to read ‘Goodbye Hal’, what is the SQL code necessary to execute the HELLO_WORLD procedure?

Execute the procedure with your code. What is printed to the DBMS Output pane?

4d Execute the HELLO_WORLD procedure, passing it the following parameters:

Greeting: **Hello**

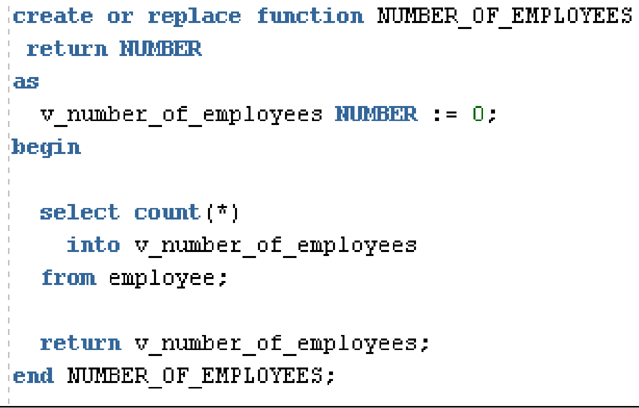
Name: **null**

What is printed to the DBMS Output pane?

Creating a (parameter-less) Function

0.5 pt

5 Run the following PL/SQL code to create a function named NUMBER_OF_EMPLOYEES:

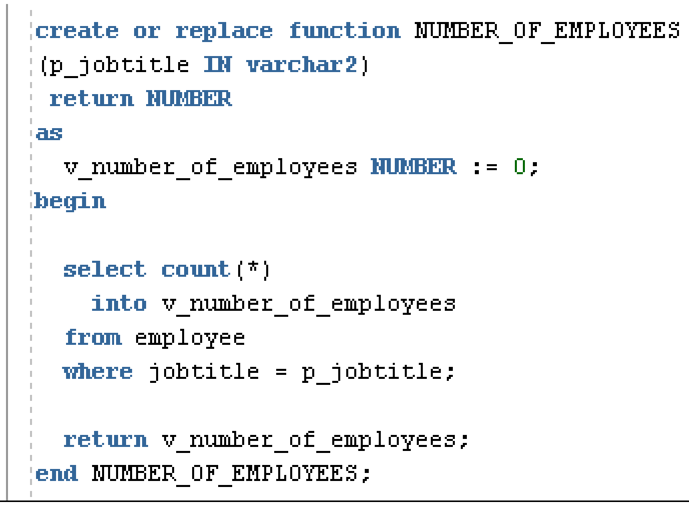


```
create or replace function NUMBER_OF_EMPLOYEES
 return NUMBER
as
  v_number_of_employees NUMBER := 0;
begin

  select count(*)
    into v_number_of_employees
  from employee;

  return v_number_of_employees;
end NUMBER_OF_EMPLOYEES;
```

What is the SQL code necessary to execute the NUMBER_OF_EMPLOYEES function?

Execute the function with your code. Include the results in your submission.

Creating a Function that accepts a single IN parameter

<table>
<tr><td colspan="2">6b is worth 1.5pt, the rest is worth 0.5 each. 0.5 * 7 + 1.5 = 5 pts</td></tr>
<tr><td><b>6</b></td><td>Modify the NUMBER_OF_EMPLOYEES function to accept and use a single IN parameter as shown below.

<pre>
create or replace function NUMBER_OF_EMPLOYEES
(p_jobtitle IN varchar2)
 return NUMBER
as
  v_number_of_employees NUMBER := 0;
begin

  select count(*)
    into v_number_of_employees
  from employee
  where jobtitle = p_jobtitle;

  return v_number_of_employees;
end NUMBER_OF_EMPLOYEES;
</pre></td></tr>
<tr><td><b>6a</b></td><td>If we wanted to determine the number of engineers (job title “Engineer”) that work at Eagle Electronics using the NUMBER_OF_EMPLOYEES function, what is the SQL code necessary to execute the function?
Execute the function with your code. Include the results in your submission.</td></tr>
<tr><td><b>6b</b></td><td>Modify the NUMBER_OF_EMPLOYEES function to make it more robust. The search by job title should:
• be case in-sensitive (e.g., it should ignore capitalization differences)
• ignore differences resulting from leading and trailing spaces
• change the data type of the parameter to be based upon the JOBTITLE column of the EMPLOYEE table

Include the updated function definition in your submission.</td></tr>
<tr><td><b>6c</b></td><td>Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title:
Job title: <b>engineer</b>
Verify your solution works by running it. Include the results in your submission.</td></tr>
<tr><td><b>6d</b></td><td>Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title:
Job title: <b>dba</b>
Verify your solution works by running it. Include the results in your submission.</td></tr>
<tr><td><b>6e</b></td><td>Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title:
Job title: <b>DBA</b>
Verify your solution works by running it. Include the results in your submission.</td></tr>
<tr><td><b>6f</b></td><td>Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title:
Job title: <b>chief sales officer</b>
Verify your solution works by running it. Include the results in your submission.</td></tr>
<tr><td><b>6g</b></td><td>Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title:
Job title (with 3 leading and trailing spaced): <b>‘   chief sales officer   ’</b></td></tr>
</table>

	Verify your solution works by running it. Include the results in your submission.
6h	Use the updated NUMBER_OF_EMPLOYEES function to determine the number of employees possessing the following job title: Job title: **CEO** Verify your solution works by running it. Include the results in your submission.
Writing your own code (Total: 6.5 pts)	
7a (2 pts)	Create a **function** named **DAYS_AWAY** that accepts a single parameter of type DATE. • If the date is in the future, the function should return a positive whole number reflecting the number of days in the future. • If the date is in the past, the function should return a negative whole number reflecting the number of days in the past. When performing calculations do NOT hard code a date into your logic. The current date should be dynamically determined at run-time. Hence the output of the function will be different if executed tomorrow than if it was executed today!
7b (0.5 pt)	Verify your DAYS_AWAY function by passing in **a future date** and **a past date**. Include both the code necessary to execute the function, and the resulting output in your submission.
8a (1.5 pt)	Create a **procedure** named **DAY_OF_THE_WEEK** that accepts a single parameter of type DATE. The procedure should determine the day of week, and output the following: • The day of the week that the provided date (e.g., the argument) falls upon • The day of the week for the date immediately prior to the argument • The day of the week for the date immediately **after** the argument The day of week should be in string format, i.e. Tuesday, but not a number i.e. 3 Your logic should never use a hard-coded date or value. All day of week or date functionality should be dynamically determined.
8b (0.5 pt)	Verify your DAY_OF_THE_WEEK procedure by passing in **a random date** and **SYSDATE**. Include both the code necessary to execute the procedure, and the resulting output in your submission.
8a (1.5 pt)	Create a procedure **DAYS_FROM_WEEKEND**. This function accepts a single parameter of type DATE. If no input is provided, assumes the date is the current date. • If the day of the week is on weekend, print 'Happy Weekend!' • Otherwise, print the number of days from the date to the next Saturday. Please do not hard code the date.
8b (0.5 pt)	Verify your **DAYS_FROM_WEEKEND** procedure by passing in **a random date** and **SYSDATE**. Include both the code necessary to execute the procedure, and the resulting output in your submission.

Table 3: Procedures and Functions (More Fun, 18.9 pts)

Creating a (parameter-less) Procedure (2 + 0.2 + 0.5 = 2.7 pts)	
1a	Using Eagle database, create a **procedure** that will count a number of suppliers.
1b	Execute the procedure. Include the results in your submission.
1c	What is the difference between anonymous blocks and named blocks? Here is a helpful reference: https://blogs.oracle.com/oraclemagazine/building-with-blocks
Default Coding Standards (e.g., formatting) for PL/SQL code The following method can be used to apply Oracle recommended capitalization to PL/SQL code. If desired, custom templates for formatting can be created. (0.2 pt)	
2	Highlight the code from 1a. Right click and then select: **FORMAT** What is the result? Include the formatted code in your answer template.
Creating a Procedure that accepts a single IN parameter (2 + 0.2 = 2.2 pts)	
3a	Modify your procedure from 1a to take a city as input. If the city doesn't exist, please tell the user. Consider upper and lower cases for input, state clearly what is acceptable and what is not.
3b	Execute the procedure. Include the results in your submission.
Creating a Procedure that accepts multiple IN parameters (2 + 0.2 + 0.2 = 2.4 pts)	
4a	Modify your procedure from 3a to take a state, in addition to city, as input. Make certain to state all your assumptions clearly.
4b	Execute the procedure. Include the results in your submission.
4c	Execute the procedure without providing a city (provide only the state). What happens? Why?
Creating a (parameter-less) Function (2 + 0.2 = 2.2 pts)	
5a	Create a **function** that will count the number of suppliers.
5b	Execute the function. Include the results in your submission.
Creating a Function that accepts a single IN parameter (2 + 0.2*6 + 0.5 = 3.7 pts)	
6a	Modify your function from 5a to take city as input.
6b	Execute the function and pass in the following parameters • Dalton • SouthBend • South Bend • dalton • DALTON • Daton (this is not a typo) Include the results for each of the parameter in your submission.
6c	Depending on your logic in 6a, you may or may not get results for the parameters in 6b. Explain what you would change in order for your code to provide numbers for each of them. You Do NOT need to modify your code.
Creating a "Simple" Loop in PL/SQL (2 + 0.2 = 2.2 pts)	
7a	Create a **procedure** that would count a number of suppliers with a zipcode that starts with a particular digit.

	• Your loop should be from 0 to 9 • For each digit, please output the number of suppliers and the number of zipcodes that start with that digit.
7b	Execute your procedure. Include the results in your submission.
Creating a WHILE Loop in PL/SQL (1.5 + 0.2 = 1.7 pt)	
8a	Modify the procedure in 7a to use a WHILE loop instead.
8b	Execute your procedure. Include the results in your submission.
Creating a FOR Loop in PL/SQL (1.4 + 0.2 = 1.6 pt)	
9a	Modify your procedure in 8a to use a FOR loop instead.
9b	Execute your procedure. Include the results in your submission.