**LAB 06**

**Name:** Shani Patel

**Seneca ID:** 152243192

**Seneca email:** [sgpatel22@myseneca.ca](mailto:sgpatel22@myseneca.ca)

**Subject code:** NBB

**Submission date:** 16/07/2021

1. Write a store procedure called *Get\_Fact* that gets an integer number *n* and calculates and displays its factorial.

Example:

0! = 1  
2! = fact(2) = 2 \* 1 = 1  
3! = fact(3) = 3 \* 2 \* 1 = 6  
. . .  
n! = fact(n) = n \* (n-1) \* (n-2) \* . . . \* 1

**Show your testing with 2 different integers and capture screenshot.**

CREATE OR REPLACE PROCEDURE Get\_Fact(n INTEGER) AS

factorial INTEGER := 1;

BEGIN

FOR i IN REVERSE 1..n LOOP

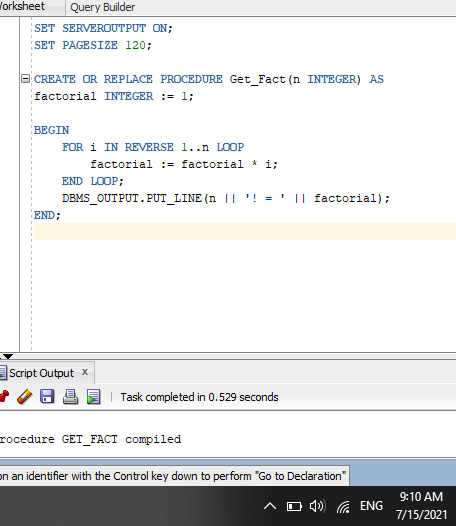
factorial := factorial \* i;

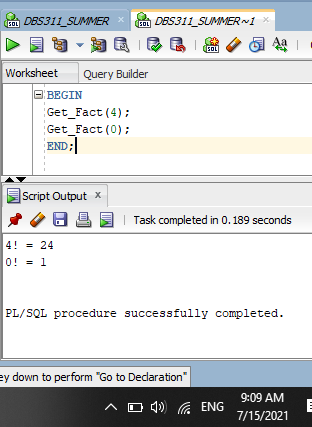
END LOOP;

DBMS\_OUTPUT.PUT\_LINE(n || '! = ' || factorial);

END;

/

**



1. The company wants to calculate the employees’ annual salary:

The first year of employment, the amount of salary is his/her base salary (shown under column Salary).

Every year after that, the salary increases by 5%.

Write a stored procedure named *Calculate\_Salary* which gets an Employee ID and for that employee calculates the salary based on the number of years the employee has been working in the company. (Use a loop construct to calculate the salary).

The procedure calculates and prints the Name and Annual Salary.

Sample output:

First Name: first\_name

Last Name: last\_name

Annual Salary: $99,999

If the employee does not exist, the procedure displays a proper message.

**Show your testing with an invalid ID and the other one with valid ID and capture screenshot.**

CREATE OR REPLACE PROCEDURE Calculate\_Salary (empID employee.employee\_id%type) AS

emp employee%rowtype;

newSalary employee.salary%type;

yearsWorked INTEGER;

BEGIN

SELECT \* INTO emp

FROM employee

WHERE employee\_id = empID;

newSalary := emp.salary; --starting salary

yearsWorked := trunc(MONTHS\_BETWEEN(SYSDATE, emp.hire\_date) / 12); --trunc so we dont round up an extra year

FOR year IN 1..yearsWorked LOOP

newSalary := newSalary \* 1.05;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('First Name: ' || emp.first\_name);

DBMS\_OUTPUT.PUT\_LINE('Last Name: ' || emp.last\_name);

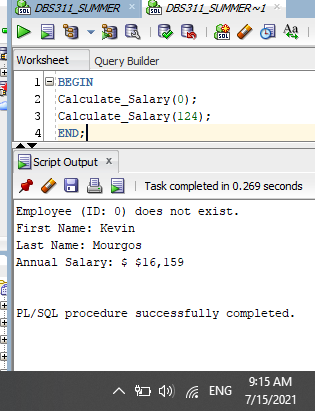
DBMS\_OUTPUT.PUT\_LINE('Annual Salary: $' || to\_char(newSalary, '$99,999');

EXCEPTION WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Employee (ID: ' || empID || ') does not exist.');

END;

/

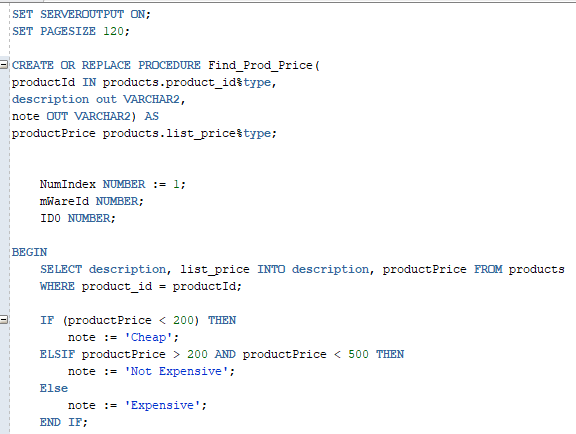


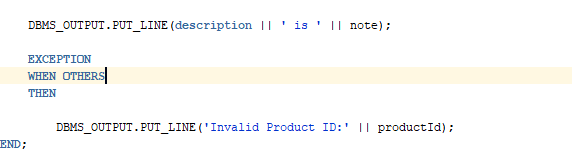
1. Write the code for the procedure called *Find\_Prod\_price*, that will search table Products and for a given Product ID will find its Description and display a message (note) regarding its List Price. This note will show *Cheap* for price below $200, *Not Expensive* for price between $200 and $500, otherwise will be *Expensive* (for price higher than $500). You need to take care of the wrong input (Product ID is invalid) as well.

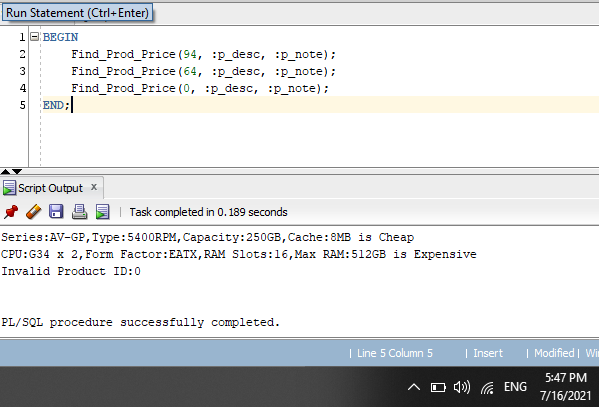
Use one IN parameter and two OUT parameters, then use PL/SQL block to show your output like (for a given ID of 31):

CPU:LGA2011-3 x 2,Form Factor:EATX,RAM Slots:16,Max RAM: is Not Expensive

**Show your testing with a Cheap, Expensive and Invalid product and capture screenshot.**







4. Write a stored procedure named *Warehouses\_Report* to print the warehouse ID, warehouse name, and the city where the warehouse is located in the following format for ALL warehouses:

Warehouse ID:

Warehouse name:

City:

State:

If the value of state does not exist (null), display “no state”.

The value of warehouse ID ranges from 1 to 9.

You can use a loop to find and display the information of each warehouse inside the loop.

(Use a loop construct to answer this question. **Do not use cursors**.)

Capture screenshot of your code and successful run with output.

