# DBS311A Lab 6 *– E-Mail submission only, your document must have both CODE content and the OUTPUT after execution of the Procedure.*

# Subject of your mail must be like 311-Lab6 by Smith, John

# *Due by Saturday, March 20th by 9pm*

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.**

**set serveroutput on;**

**create or replace procedure Get\_Fact(n int)**

**as**

**factorial int :=1;**

**i int :=0;**

**begin**

**loop**

**factorial := (factorial\*(n-i));**

**i:=i+1;**

**exit when i= n;**

**end loop;**

**dbms\_output.put\_line('factorial: ' || factorial);**

**exception**

**when others then**

**dbms\_output.put\_line('error occured! ');**

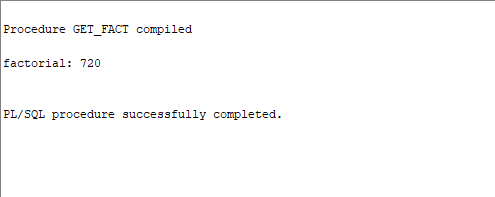
**end;**

**execute Get\_Fact(6);**

**Execution 1:**

**execute Get\_Fact(6);**

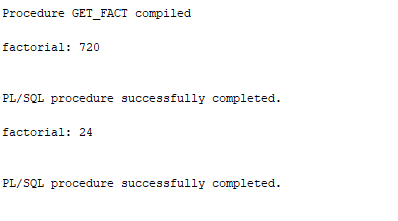
**Output:**



**Execution 2:**

**execute Get\_Fact(4);**

**Output :**



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.

set serveroutput on;

create or replace procedure Calculate\_Salary(m\_employee\_id number)

is

m\_last\_name employee.last\_name % type;

m\_first\_name employee.first\_name % type;

m\_salary employee.salary % type;

m\_hire\_date employee.hire\_date % type;

m\_year number;

i number:=0;

begin

select

first\_name, last\_name, hire\_date, salary into

m\_first\_name,m\_last\_name,m\_hire\_date,m\_salary

from employee

where employee\_id = Calculate\_Salary.m\_employee\_id;

select trunc(to\_char(sysdate - hire\_date)/365) into m\_year

from employee

where employee\_id = Calculate\_Salary.m\_employee\_id;

loop

m\_salary := m\_salary \* 1.05;

i:=i+1;

exit when i = m\_year;

end loop;

dbms\_output.put\_line('First Name: '|| m\_first\_name);

dbms\_output.put\_line('Last Name: '|| m\_last\_name);

dbms\_output.put\_line('Salary: $'|| m\_salary);

exception

when no\_data\_found then dbms\_output.put\_line ('Employee id not found !');

when others then dbms\_output.put\_line('Error Occured !');

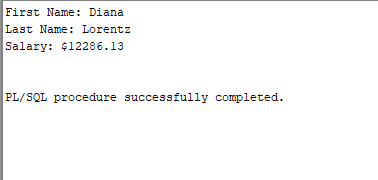
end;

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

**Valid Test:**

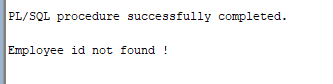
**execute Calculate\_Salary(107);**

**Output:**



**Invalid Test:**

**execute Calculate\_Salary(1999);**



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.**

**CREATE or replace PROCEDURE Find\_Prod\_price(productID IN NUMBER, productDescription OUT VARCHAR2, productType OUT VARCHAR2)**

**IS**

**number\_of\_record NUMBER;**

**BEGIN**

**SELECT**

**COUNT(\*) INTO number\_of\_record FROM Products WHERE PRODUCT\_ID = ProductID;**

**SELECT**

**DESCRIPTION,**

**CASE**

**WHEN LIST\_PRICE < 200**

**THEN 'Cheap'**

**WHEN LIST\_PRICE <= 200 AND LIST\_PRICE >= 500**

**THEN 'Not Expensive'**

**ELSE**

**'Expensive'**

**END**

**INTO productDescription, productType**

**FROM PRODUCTS**

**WHERE PRODUCT\_ID = find\_prod\_price.productID;**

**EXCEPTION**

**when no\_data\_found then dbms\_output.put\_line ('Product id not found !');**

**when others then dbms\_output.put\_line('Error Occured !');**

**END;**

**One with the expensive :**

**DECLARE**

**productDescription VARCHAR2(2000); productType VARCHAR2(50);**

**BEGIN**

**Find\_Prod\_price(31, productDescription, productType);**

**IF(productDescription IS NOT NULL)**

**THEN**

**dbms\_output.put\_line(productDescription || ': is ' || productType);**

**END IF;**

**END;**

**One with the cheap:**

**DECLARE**

**productDescription VARCHAR2(2000); productType VARCHAR2(50);**

**BEGIN**

**Find\_Prod\_price(233, productDescription, productType);**

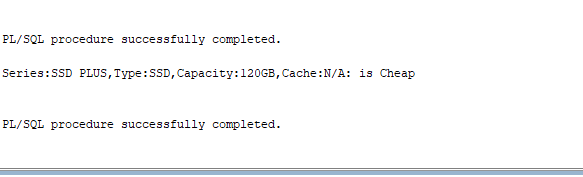
**IF(productDescription IS NOT NULL)**

**THEN**

**dbms\_output.put\_line(productDescription || ': is ' || productType);**

**END IF;**

**END;**

**OUTPUT:**

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**.)

create or replace procedure Warehouse\_Report

is

m\_warehouse\_id warehouses.warehouse\_id %type;

m\_warehouse\_name warehouses.warehouse\_name %type;

m\_city locations.city % type;

m\_state locations.state %type;

begin

FOR i IN 1..9 loop

select w.warehouse\_id, w.warehouse\_name, l.city, nvl(l.state,'no state') into

m\_warehouse\_id, m\_warehouse\_name, m\_city, m\_state

from warehouses w inner join locations l

on w.location\_id = l.location\_id

where

w.warehouse\_id = i;

dbms\_output.put\_line('Warehouse ID: ' || m\_warehouse\_id);

dbms\_output.put\_line('Warehouse name: '|| m\_warehouse\_name);

dbms\_output.put\_line('City: '|| m\_city);

dbms\_output.put\_line('State: ' || m\_state);

dbms\_output.put\_line(' ');

end

loop;

exception

when others then dbms\_output.put\_line('Error Occured');

end;

execute warehouse\_report();

OUTPUT:

