**COMP255**

**Lab 8**

**SQL Substitution Variables, Procedures and Functions**

**Due Date:** Tuesday, December 5thby midnight

**Points**: 20

1. Create a connection and log into Oracle using SQL Developer. Open up a text editor such as notepad and enter the following in the top of the file (dashes indicate a comment):

-- Your Name

-- COMP255

-- Lab 8

As before, when you create your commands for this lab, paste a copy into the text file. You will submit the text file into the D2L dropbox. In your submission, show both the procedures/functions you build and the command you use to test them.

1. We will be using the same data tables as we used for our lab 7. If you want to recreate them, paste the commands from the script into the SQL worksheet and rerun them. Then clear your worksheet. Do not include the build script commands in your submission.
2. Using our class notes on SQL substitution variables, procedures and functions, create SQL for the following:
3. (2) Show all employees displaying employee id, last name and title for a given department. Use a substitution variable to input the department id.
4. (2) Show all employees displaying employee id, last name and title for a given department name. Use a substitution variable to input the department name.
5. (2) Show all employees by ID, first name, last name, hiredate, salary who make less than a given salary and were hired on or after given starting date but before a given ending date. Use substitution variables for the starting date, ending date and the salary.
6. (2) Create a stored procedure that updates an employee record if the employee gets married. Call the procedure PRC\_UPDATE\_EMP\_NOTMARRIED. It should take one parameter – the employee id and should set the marital status to indicate the employee is not married. Do not use a substitution variable for any of the functions or procedures. Call the procedure, checking the data before and after the call. Show your test call.
7. (3) Create a stored procedure to give all employees with a certain title a raise of a given percentage. Call the procedure PRC\_APPLY\_RAISE. It should take two parameters – one for the title and one for the percentage (use a decimal value such as 0.05). Again do not use substitution variables. Call the procedure, checking the data before and after the call. Show your test call.
8. (3) Create a function to determine the projected compensation based for an employee based on a percentage raise and a given bonus. The employee id, percentage and bonus should all be parameters to the function. Call the function FTN\_COMPUTE\_TOTAL\_COMPENSATION. Use a decimal for the raise. The bonus will just be a number – e.g. 200 for 200 dollars. The function should not change the table data – just return the computed compensation. Test the function using:

SELECT 3, FTN\_COMPUTE\_TOTAL\_COMPENSATION(3, 0.05, 1025) FROM dual;

This test will show what the result would be for employee 3. (Selecting 3 up front just shows the employee number.)

1. (3) Create a function to determine the total payroll based on job title. The function should take one parameter – the title – and sum up all the salaries for that title and return the resulting value. Call it FTN\_PAYROLL. To test, use a select statement to show these employees for each title – i.e. grouped by title. (See the example in our notes.)
2. (3) Create a function to determine the average salary for all employees hired after a given date and assigned to a given department. The function should take two parameters – the date and the department number. Call it FTN\_COMPUTE\_AVGSAL. Use a select statement to show the average for each department and for a specific hire date. You may choose the date you want to check. Just hard code the test date in the call. Look at the table to find appropriate values.