**COEN 178 Intro to Database Systems Fall 2019**

**Lab 5 (75 pts)**

**Objectives: Learn**

* PLSQL Functions
* Exceptions

In this lab, we will use **AlphaCoEmp** and **Emp\_Work** tables that you have created in the earlier labs.

**The following is the plan of what you will do today.**

* + - * Load randomly generated values for salary and title for the tuples in **AlphaCoEmp** using a procedure.
      * Write a function called **calcSalaryRaise()** to calculate the amount of raise for an employee.
      * Create a table called **EmpStats**.
      * Create a procedure called **saveCountsByTitle**
      * Create a function called **countByTitle() with an exception**

Run the queries and capture the results in **lab5\_output.lst**, using *spool*.

**Question 1 (10 pts)**

In this exercise, we will try to assign job titles to employees randomly selected from a hardcoded set of titles, in AlphaCoEmp table, using a **PLSQL procedure**. Since we want to assign salaries based on titles, we will store titles and salaries in two **PLSQL VArrays** and randomly select an index into the arrays and use the title and its associated salary from the same index number. For example, the salary for the title stored at index 0 in the **titles array** will be at index 0 in the **salaries array**.

Note: In PLSQL VArrays, subscripts start at 1, not 0.

Examine the code in the procedure below, to assign job titles and salaries.

**Create or Replace Procedure assignJobTitlesAndSalaries**

**As**

**type titlesList IS VARRAY(5) OF AlphaCoEmp.title%type;**

**type salaryList IS VARRAY(5) of AlphaCoEmp.salary%type;**

**v\_titles titlesList;**

**v\_salaries salaryList;**

**Cursor Emp\_cur IS**

**Select \* from AlphaCoEmp;**

**l\_emprec Emp\_cur%rowtype;**

**l\_title AlphaCoEmp.title%type;**

**l\_salary AlphaCoEmp.salary%type;**

**l\_randomnumber INTEGER := 1;**

**BEGIN**

/\* Titles are stored in the v\_titles array \*/

/\* Salaries for each title are stored in the v\_salaries array.

The salary of v\_titles[0] title is at v\_salaries[0].

\*/

**v\_titles := titlesList('advisor', 'director', 'assistant', 'manager', 'supervisor',’secretary’);**

**v\_salaries := salaryList(130000, 100000, 600000, 500000, 800000, 90000);**

**/\* use a for loop to iterate through the set**

**for l\_emprec IN Emp\_cur**

**LOOP**

**/\* We get a random number between 1-5 both inclusive \*/**

**l\_randomnumber := dbms\_random.value(1,5);**

**/\* Get the title using the random value as the index into the**

**v\_tiles array \*/**

**l\_title := v\_titles(l\_randomnumber);**

**/\* Get the salary using the same random value as the index into the v\_salaries array \*/**

**l\_salary := v\_salaries(l\_randomnumber);**

**/\* Update the employee title and salary using the l\_title**

**and l\_salary \*/**

**update AlphaCoEmp**

**set title = l\_title**

**where name = l\_emprec.name;**

**update AlphaCoEmp**

**set salary = l\_salary**

**where name = l\_emprec.name; NAME TITLE CURRENTSALARY NEWSALARY**

**------------------------- -------------------- ------------- ----------**

**Stone 71850 1437**

**END LOOP;**

**commit;**

**END;**

**/**

**Show errors;**

Run the code and if it compiles without errors, run the command,

1. a) exec **assignJobTitlesAndSalaries** at SQL prompt.
2. b) Run a Select \* on AlphaCoEmp table and check if titles and salaries are assigned.
3. c) Now, modify the above procedure and include one more job title and a salary for that title in the code.
4. d) Run the procedure.
5. e) Execute the procedure and make sure it is working ok.

**Question 2 (15 pts)**

You will write a **PLSQL function** that **calculates the salary raise based on the current salary and the percent raise**. PLSQL functions return a value.

The function **calcSalaryRaise()** calculates raise amount as follows:

• **Takes the name and percent salary (an integer) as parameters.**

**• Fetches the employee’s salary from AlphaCoEmp Table and calculates the amount of raise.**

**Create or Replace Function calcSalaryRaise( p\_name in AlphaCoEmp.name%type, percentRaise IN Integer)**

**RETURN NUMBER**

**IS**

**l\_salary AlphaCoEmp.salary%type;**

**l\_raise AlphaCoEmp.salary%type;**

**l\_cnt Integer;**

**BEGIN**

-- Find the current salary of p\_name from AlphaCoEMP table.

**Select salary into l\_salary from AlphaCoEmp**

**where name = p\_name;**

-- Calculate the raise amount

**l\_raise := l\_salary \* (percentRaise/100);**

**/\* return a value from the function \*/**

**return l\_raise;**

**END;**

**/**

**Show Errors;**

Run the function (copy and paste it at SQL prompt or run it from a script file). Once it compiles without errors, you can execute it by calling it, as shown below.

1. a) **If you want to test the function and see if it is working ok, call it as follows:**

**Select calcSalaryRaise('Stone',2) from Dual;**

Note: You can give any name that is in the AlphaCoEmp table.

1. b) Call the function as part of a more useful SQL query

**Select name, title, salary CURRENTSALARY, trunc(calcSalaryRaise(name,2)) NEWSALARY**

**from AlphaCoEmp where upper(name) = upper('Stone');**

What is the output?------------------------

1. c) If you examine the code of the function, we are comparing (string compare) the name with the parameter, p\_name without checking the case. Modify the code so that both strings are compared with each other, both in upper or lowercase.

Test and make sure your function work correctly after modifications.

d) What happens if you call the function with a name that NOT in the AlphaEoEmp table?

**Select calcSalaryRaise(*'Choose a name that is not in the AlphaCoEmp* ',2) from Dual;**

Now let us rewrite the function with an exception.

Add the given code below the “return l\_raise” statement.

**EXCEPTION**

**WHEN NO\_DATA\_FOUND THEN**

**dbms\_output.put\_line('No employee with given name found!');**

**--return 0;**

**WHEN others THEN**

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

Test your function again with a name that is not in the table.

**Question 3 (15 pts)**

1. a) Let us create a table called **EmpStats** as follows:

**Create table EmpStats (title VARCHAR(20) Primary KEY,empcount INTEGER, lastModified DATE);**

1. b) The function (incomplete) below, **counts the number of employees from AlphaCoEmp table by title, where title is passed as a parameter and returns the count**.

**Complete the function and run it.**

**Create or Replace Function countByTitle(p\_title in AlphaCoEmp.title%type)**

**RETURN NUMBER IS**

**l\_cnt Integer;**

**BEGIN**

**/\* Complete the query below \*/**

**Select into l\_cnt from AlphaCoEmp**

**Group by**

**Having**

**return l\_cnt;**

**END;**

**/**

1. **c) Run the SQL commands below and show the output.**

**select countByTitle('director') from Dual;**

**select countByTitle('advisor') from Dual;**

**Question 4 (15 pts)**

If the function, **countByTitle()** in Q3 is working ok, we will write a procedure to store the number of employees by title in the table, EmpStats. We will call the function countByTitle() in this procedure.

**CREATE or REPLACE procedure saveCountByTitle**

**AS**

**l\_advisor\_cnt integer := 0;**

**BEGIN**

l\_advisor\_cnt := countByTitle('advisor');

**delete from EmpStats; -- Any previously loaded data is deleted**

**/\* inserting count of employees with title, ‘advisor’.\*/**

**insert into EmpStats values ('advisor',l\_advisor\_cnt,SYSDATE);**

**END;**

**/**

**Show errors;**

The above procedure stores the count of employees with the title, ‘advisor’.

1. a) Complete the procedure to store the count of employees for every title you have in AlphaCoEmp table.
2. b) Execute the procedure.
3. c) Show the data in the EmpStats table (do a Select \* ).

**Question 5 (20 pts)**

1. Write a **PLSQL function** called **countBySalaryRange()** that takes two parameters, called **lowEnd** and **highEnd**, both numbers. The function counts the no.of employees in AlphaCoEmp table, whose salaries fall with in the range between lowEnd and highEnd (both numbers inclusive) and returns the count.
2. Check if your function is created without errors.
3. Now, call the function, countBySalaryRange() with two values of your choice for the parameters and display the value returned.
4. Show how you call countBySalaryRange() with two values of your choice for the parameters and display the value returned with an appropriate column heading.