COEN 178 **Intro to Database Systems** **Spring 2018**

Lab 5 (100 pts)

Objectives: Learn

- PLSQL Functions
- PLSQL Triggers
- PHP and SQL

In this lab, we will use **AlphaCoEmp** and **Emp_Work** tables that you have created in the earlier labs. You may need to create 2 new tables as we follow through the questions.

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

a) Load randomly generated values for salary and title for the tuples in **AlphaCoEmp** using a procedure.

b) Write a function called **calcSalaryRaise()** to calculate the amount of raise for an employee.

c) Create a table called **EmpStats**.

d) Create a procedure called **saveCountsByTitle**

e) Create a function called **countByTitle()**

f) Write a trigger to change the count by title

g) Write a PHP program with SQL to access the database and display the results.

Run the queries and capture the results in **lab5_output.lst**, using *spool*. Turn in your modified procedures/functions from parts 1, 2, 3, 4, and 6 in a sql file and your lab5_output.lst. Demo question 7 during lab.

Part 1

Question 1 (15 pts)

You have used a function to generate a random number and used it to populate the salaries in AlphaCoEmp table.

Today, we will try to assign job titles to employees randomly selected from a hardcoded set of titles. Since we want to assign salaries based on titles, we will store titles and salaries in two PLSQL arrays 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.

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
     v_titles := titlesList('advisor', 'director', 'assistant',
'manager', 'supervisor');

     v_salaries := salaryList(130000, 100000, 600000, 500000,
800000);

     for l_emprec IN Emp_cur
     loop
          l_randomnumber := dbms_random.value(1,5);

          l_title := v_titles(l_randomnumber);
          l_salary := v_salaries(l_randomnumber);

           update AlphaCoEmp
           set title = l_title
           where name = l_emprec.name;
```

```
            update AlphaCoEmp
            set salary = l_salary
            where name = l_emprec.name;

   END LOOP;
 commit;

END;

/

Show errors;
```

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

a) **exec assignJobTitlesAndSalaries** at SQL prompt.

b) Run a Select * on AlphaCoEmp table and check if titles and salaries are assigned.

c) Now, modify the above procedure and include one more job title and a salary for the title in the code.

d) Run the procedure.

e) Execute the procedure and make sure it is working ok.

Question 2 (15 pts)

Now, we 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.
- If that employee is also is in Emp_Work table, adds an additional 1000 to the raise.

```
Create or Replace Function calcSalaryRaise( p_name in
AlphaCoEmp.name%type, percentRaise IN NUMBER)
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);

     -- Check if the p_name is in Emp_Work table.
     -- If so, add a $1000 bonus to the
     -- raise amount

     Select count(*) into l_cnt from Emp_Work
     where name = p_name;

     if l_cnt >= 1 THEN
          l_raise := l_raise + 1000;
     End IF;
```

```
    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.

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.

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

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?----------

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 wither in upper case or lowercase.

Test and make sure your function is working correctly after modifications.

Question 3 (15 pts)

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

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

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

     Select into l_cnt from AlphaCoEmp
     Group by
     Having

     return l_cnt;

END;
/
```

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 the 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'.

a) **Complete the procedure to store the count of employees for <u>every</u> title you have in AlphaCoEmp table.**

b) **Execute the procedure.**

c) **Show the data in the EmpStats table (do a Select *).**

What is the count shown for the title, 'advisor'? ----------

Question 5 (15 pts)

A **trigger** is a named PL/SQL unit that is stored in the database and can be invoked automatically.

- You cannot explicitly invoke a trigger.
- A trigger fires automatically when a triggering event occurs.
- You can enable and disable triggers.
- You can drop a trigger you created.

We will **write a trigger that fires when the count of employees by title changes in the AlphaCoEmp table.** For example, if we insert a new row into that table with a title (from the set of titles we have used), the count by title should be changed in the EmpStats table.

```
CREATE Or Replace TRIGGER titlecountchange_trig
    AFTER INSERT ON AlphaCoEmp
    FOR EACH ROW
    BEGIN
          Update EmpStats
          set Empcount = Empcount + 1, lastmodified = SYSDATE
          where title = :new.title;
 END;
/
Show errors;
```

Run the code (either copy and paste the above code at SQL prompt. Or run it from a script file) for the above trigger.

Note the update statement where we are updating more than one column.

Once the trigger compiles without any errors, do the following at the SQL prompt,

```
insert into AlphaCoEmp Values('mickeymouse','advisor',null);
```

Now, do a query (Select *) to show the data in EmpStats table?

What is the count shown for 'advisor' title? ----------

Do you think the trigger fired? Why?

Question 6 (15 pts)

Now, we realize that the count by title in the **EmpStats** table should be automatically decremented when an employee is deleted from the **AlphaCoEmp** table.

We will write a trigger that gets fired after an insertion or a deletion into **AlphaCoEmp** table.

Let us first drop the trigger created earlier. At the SQL prompt,

```
drop trigger titlecountchange_trig;
```

```
CREATE Or Replace TRIGGER countchange_trig
    AFTER INSERT or DELETE ON AlphaCoEmp
    FOR EACH ROW
    BEGIN
     IF DELETING THEN
          Update EmpStats
          set Empcount = Empcount -1, lastmodified = SYSDATE
          where title = :old.title;
     END IF;
     If Inserting THEN
          Update EmpStats
          /* Complete the query here */
          -------------------
          where title = :new.title;
     End IF;
 END;
```

```
/
Show errors;
```

a) Complete the code in the trigger where indicated and run the trigger.

b) Now delete the row from AlphaCoEmp table where the name = 'mickeymouse'.

c) Did the trigger fire? Show reasons for your answer.

Part 2

Question 7 (10 pts)

In this, you will run PHP program that will connect to your Oracle database tables, fetches and shows the data.

Copy and paste the program below into a file called showEmps.php and place the file in the folder where you are required to put the .php files.

You must edit the code below to put your login and password. At the end of the lab session, please feel free to change your password.

```
<?php
//connect to your database
// You must edit the line below to give your user name, password
// and a correct path to your database

$conn=oci_connect('username','password', 'localhost/XE');
if(!$conn) {
          print "<br> connection failed:";
        exit;
}
```

```
//    Parse the SQL query
$query = oci_parse($conn, "SELECT name,title FROM AlphacoEmp Order
by name");

// Execute the query
oci_execute($query);

// Prepare to display results
echo "<b>";
// Fetch each row. the first column is 0, then 1, etc.
while($row=oci_fetch_array($query)){
     // String compare
     if (strcmp($row[1],'advisor') == 0){
          echo "<font color='green'> $row[0] </font></br>";
     }
     else{
          echo "<font color='blue'> $row[0] </font></br>";
     }
}
echo "</b>";

// Log off
OCILogoff($conn);
?></html>
```

In your browser window, enter the URL for the script.

Did you see the output?