PROJECT Due: April 26, 2019 Friday

Group Assignment (You may form groups up to two individuals in one group)

1. For this project, you need to create the company database and populate it with the data given in the Oracle SQL PLUS environment (if you have not done so)
   (a continuation of homework #7).
2. Your SQL commands should be placed in file(s) and run from the file(s).
3. Implement Triggers on all the tables to change character data to uppercase in the character data type columns.
4. Implement a Function that returns Dname from Department for any given employee.
   The parameter to the function would be the SSN.
5. Implement a Function that returns manager's full name for any given department.
   The parameter to the function would be the department name.
6. Implement another Function such that for any given project, it should return full name of the manager of the department that controls the project. The parameter to the function would be the project name.
7. Implement a Procedure that increases an employee salary by x%. Employee is identified by SSN and the percentage of increase is given as an input.
8. Implement a Package that contains the following methods:
   Count the number of dependents for any given valid employee (test first if the employee is a valid one).
   Add a dependent for any given valid employee (test first if the employee is a valid one).
   Remove a dependent for any given valid employee (test first if the employee is a valid one).

Partial solution: `K:\Courses\2019-Spring\ITC341-ugur1a\_Materials_\proj-partial\`

In addition to the code for your triggers, functions, procedures, and packages, you need to include the code to demonstrate that all of your triggers, functions, procedures, and packages work correctly. You need to submit the entire code and screen captures of the execution results (i.e., the spool file) similar to previous homework assignments (using spool command in SQL-Plus).

Also, you need to submit a project report (a Word or text file or similar document). The **project report** should include:
- Name of group members, task distribution (i.e., who has done what)
- Source code, screen logs using spool command.
- Technical problems that you encounter during the implementation and your solution.
- What have you learned from this project, your suggestion to the course and project.

All of your **project.sql**, **project-out.txt** files and the **project-report** should be submitted electronically.

What to hand in:

After testing your PL/SQL subprograms and completing the report, create a folder 'project' and put all of your files in the folder.

1. First make sure that the results are correct (i.e., testing needs to be completed first).
2. **Submit the entire `project` folder.** Copy the entire **`project`** folder into the Submit drive **K:\Courses\2019-Spring\ITC341-ugur1a\22363520\yourCMUglobalId** folder.
3. Zip your project folder as **yourCMUid–proj.zip** and submit to the BlackBoard as well.

   One submission per group is OK. The other member of the group needs to create a text file called "**yourCMUid-mynotes.txt**" and put info about the name of his/her project partner in the file, and submit the **yourCMUid-mynotes.txt** only (no folder) into both the the K drive and BlackBoard.

Some useful SQL-Plus commands you need to use before the final run: (Sample Code is given in page 2.)

```
SQL>  set serveroutput on      --  this is a MUST (for displaying stored procedure – function results)
SQL>  set echo on              --  this is a MUST (for displaying queries in the output file)
SQL>  set linesize 120         --  this is NEEDED - for displaying larger tables
```

```
SQL>  set timing on
```
-- **this is NEEDED** - for displaying the query execution time

-- SAMPLE CODE
Also see the files in the **Partial solution:** `K:\Courses\2019-Spring\ITC341-ugur1a\_Materials_\proj-partial\`

```
set echo on
set serveroutput on
set linesize 120
set timing on

create or replace trigger project_trigger
after insert on project
begin
       UPDATE project SET  pname = upper(pname), plocation = upper(plocation);
end;
/

-- function example
CREATE OR REPLACE FUNCTION get_dept(EMPSSN IN VARCHAR2)
   RETURN varchar2
IS
   mm varchar2(15);
BEGIN
   SELECT dname into mm FROM EMPLOYEE E ,DEPARTMENT D
   WHERE E.DNO = D.DNUMBER AND E.SSN = EMPSSN ;

   exception
     when NO_DATA_FOUND then
       dbms_output.put_line('Employee with Id ' || empssn || ' does not exist');

   RETURN(mm);
  -- one can also return a concatenated string if necessary
   -- ex: if to return fname, lname assuming fn holds fname, lname holds lname
   -- return (fn || ' ' || ln);
END;
/
show errors

-- **** Test **** -- funtion call
declare
  deptname department.dname%type;
-- calling the function get_dept  - ssn has the type VARCHAR(9).
begin
  deptname := get_dept_name('123456789');
  dbms_output.put_line('dept name for ' || '123456789' || ' is ' || deptname);

  deptname := get_dept_name('111111111');
  dbms_output.put_line('dept name for ' || '111111111' || ' is ' || deptname);
end;
/

-- stored procedure example
CREATE OR REPLACE PROCEDURE get_dept_sp(EMPSSN IN VARCHAR2)
AS
   mm varchar2(15);
BEGIN
   SELECT dname into mm FROM EMPLOYEE E ,DEPARTMENT D
   WHERE E.DNO = D.DNUMBER AND E.SSN = EMPSSN ;

   dbms_output.put_line('Dept. name is '|| mm);

   -- exception handling
   exception
     when NO_DATA_FOUND then
       dbms_output.put_line('No data found');
END;
/
show errors

-- calling the stored procedure get_dept_sp
exec get_dept_sp('999887777');
exec get_dept_name_sp('111111111');
```