CS54100 - Project 1

Fall 2015

Due: Wednesday Sept. 30, 2015, 11:59PM

(There will be a 10% penalty for each late day up to five days. The assignment will not be accepted afterwards.)

Part1: (50 Points)

**Introduction**

Each student has to do this project individually. You are going to use Oracle to perform some queries and create views for a database. The schema and sample data of the database are provided. The project should be run on CS sun workstations.

Information about getting your Oracle account and general initial configuration is available in:

<https://www.cs.purdue.edu/resources/facilities/oracle.html>

**Your assignment**

In this project you will use the file: db.sql. Copy this file into your working directory. Create and populate tables by the following command:

SQL>@db

File db.sql will create the tables needed for this assignment. It will also fill the tables with some sample data. This will help you test your queries. The current data does not cover all the possible scenarios that the queries address and they are not completely matching the examples shown in this handout. So, feel free to add additional tuples to test some corner cases. In order to grade the assignment, the TA will be using a different data set for testing your queries. So, make sure to cover all cases.

The following section shows the schema of the database. Study the schema carefully.

**ProjectsInfo** is a database used by a software development company to keep track of its projects. ProjectsInfo keeps track of the projects, the employees/managers working on the projects including the universities they were graduated from. The back-end database of the ProjectsInfo consists of the relations defined in the following schema:

University(UnivId, UnivName)

Department(DeptId, DeptName)

Employee(EmpId, EmpName, DeptId, HomeZipCode)

Project(ProjId, ProjName)

Graduate(EmpId, UnivId, GradYear)

EmpProject(EmpId, ProjId, StartDate, EndDate)

ProjectManager(ProjId, MgrId, StartDate, EndDate)

* Relation University contains information about the universities where the employees graduated from. Attribute UnivId is the primary key.
* Relation Department contains information about the different departments in the company. Attribute DeptId is the primary key. The name of a department is unique.
* Relation Employee contains information about the employees in the company (including managers). Attribute EmpId is the primary key.
* Relation Project contains information about the projects that are running or that have been completed by the company. Attribute ProjId is the primary key. The name of a project is unique.
* Relation Graduate contains information about the university as well as the graduation year of each employee. It is assumed that each employee is graduated by one degree from one university. Thus, the primary key is defined to be EmpId.
* Relation EmpProject contains information about all the projects an employee is/was working on. The primary key is composed of the three attributes: EmpId, ProjId, StartDate as an employee can rejoin a project s/he was released from. A NULL value in the EndDate attribute indicates "Current", i.e., the employee is currently working on that project.
* Relation ProjectManager contains information about all the managers of each project. A project has only one manager at a time, but the project can have different managers at different non-intersecting time frames. A manager is identified by Attribute MgrId and references the EmpId attribute of the Employee relation. The primary key is composed of the three attributes: ProjId, MgrId, StartDate. A NULL value in the EndDate attribute indicates "Current", i.e., the manager is currently managing that project.

Queries

Write SQL queries that answer the questions below (one SQL query per question but you are allowed to use nested queries and/or the "WITH" clause of Oracle) and run them on the Oracle system. The query answers should be duplicate-free, but you should use distinct only when necessary.

If you are making any assumptions, state them clearly and document your queries.

1. Find the names of the employees as well as their graduation years who worked only on one project.
2. Find the names of the projects that have all of its employees from the same university (i.e., display Project1 if all the employees who joined Project1 were all graduated from the same university).
3. For each project, display its name as well as the number of employees who are currently working on it.
4. Find the names of the project(s) that were managed by the maximum number of different managers (if a manager managed the same project many times at different time frames, count him/her as one).
5. Find the name(s) of the university/universities that graduated the maximum number of managers.
6. For each employee, say **E**, display the name of **E** as well as the number of projects that **E** has joined (if an employee worked on the same project at different time intervals, count it as one project). Also, display the average number of days **E** has worked in a project.
7. For each project, say **P**, display **P**’s name as well as the name of the manager(s) who managed **P** for the longest period. Notice that **P** may be managed by more than one manager at different periods with the same length.
8. Find the name of the current managers who worked as non-managers before on the projects they are currently managing.

Drop

Drop all tables. Use statement “select \* from user\_catalog;” to make sure that all the objects are dropped. You can use the droptables.sql file.

What to submit

The result of your work is a file named **p1\_your\_career\_login.sql** which contains all the SQL statements you used in this assignment. Not naming your files as your **p1\_your\_career\_login.sql** may result in penalty points.

A useful strategy

Here are some useful approaches for doing the project

1. Follow the introduction about the environment setup, connect to the Oracle server with your assigned Oracle account.
2. Try a few simple SQL statements until you are comfortable interacting with sqlplus.
3. Workout the SQL statements you need to solve the above queries
4. Use a text editor you are familiar with to create a .sql file that contains the necessary SQL statements for this project.
5. Test your .sql file
6. Add the following lines in the beginning of your .sql file (**mandatory step**):

rem CS541 SQL Project 11

rem your\_first\_name your\_last\_name

rem your\_Purdue\_email\_address

1. Remember to divide and conquer. Test your.sql file continuously as you add the new SQL statements. You can use the oracle command spool to direct the output to a file that you can check. However, please do not submit the output file.

PART2 (50 points)

Description:

Writing Oracle Functions and Procedures to Process Data.

You are going to use PL/SQL (Oracle’s procedural extension to SQL) to write a few functions and procedures to process data. The result of this part of the project should be a file named **p2\_your\_career\_login.sql** that will contain all the PL/SQL statements you develop to handle the data processing tasks described below. Your grade depends on how you use the PL/SQL statements and the result of running the .sql file. You will use the same database used in PART1 of this project.

**It is required that you perform the following tasks (notice that the results shown in these exercises do not necessarily correspond to the actual values obtained from the given database):**

Create and populate all tables (This should be already done after Part1).

1. Write Procedure sp\_univ\_emp that generates information about the employees aggregated by their universities, print the output of your procedure in the following format.

UniversityName NumOfEmployees AvgProjects NumOfManagers

---------------------------------------------------------------------------------------------

Purdue 300 3.21 24

….

….

This procedure gives a list of the universities and information about employees graduated from these universities. The **NumOfEmployees** indicates the number of employees graduated from a certain university. **AvgProjects** indicates the average number of projects a graduate from the corresponding university has joined (if an employee worked on the same project at different time intervals, count it as one project). **NumOfManagers** indicates the number of employees from the corresponding university that managed at least one project at any period of time.

1. Write Procedure sp\_emp\_mate that generates for each employee, say **E**, the top two employees that worked with **E** in most of their projects. Print the output of your procedure in the following format:

Employee ID: 1

Employee Name: James

Department: Development

Graduated: 1995

EmpoyeeId EmployeeName NumOfProjects UniversityName

-------------------------------------------------------------------------------------

7 Alex 4 Maryland

2 Michael 1 Purdue

Employee ID: 2

Employee Name: Michael

Department: Development

Graduated: 2012

EmpoyeeId EmployeeName NumOfProjects UniversityName

-------------------------------------------------------------------------------------

8 Mattew 9 UCSB

9 Linda 5 UCLA

.................

This procedure generates the top two employee-mates for every employee, the ranking is based on the number of common projects the employees worked on (even if they were no time overlaps). This report is ordered by employee ID. The top employee list should be ordered based on the number of common projects as shown by the above results sample. If an employee worked on the same project at different time intervals, count that project only once.

1. Write Procedure sp\_proj\_details that generates the following report about projects:

ProjectId: 12

Project Name: HR System

ProjectStartDate NumOfParticipatedEmployees NumOfCurrentEmployees

------------------------------------------------------------------------------------------------------------

541 200 20

ManagerID StartDate Duration(Days)

-------------------------------------------------------------------

88 15-Sep-2012 150

EmpID StartDateOfIntersection EndDateOfIntersection

------------------------------------------------------------------------------------------------

98 17-Sep-2012 20-Oct-2012

99 19-Oct-2012 10-Nov-2012

ManagerID StartDate Duration(Days)

-------------------------------------------------------------

99 12-Oct-2014 220

EmpID StartDateOfIntersection EndDateOfIntersection

------------------------------------------------------------------------------------------------

101 12-Oct-2014 21-Nov-2014

105 19-Nov-2014 10-Jan-2015

Project Id: 13

Project Name: YZW

…….

This procedure generates a report about every project. For each project, say P, the report displays the Id and the name of P. It also shows the managers who managed P sorted by their start dates of management, the report shows the manager Id, start date of managing Project P as well as the number of days where Manager M managed Project P for the corresponding time slot. For each manager displayed, say M, the report displays also all the employees who were working on Project P at the time of being managed by Manager M. For each employee displayed, say E, the report shows the employee Id of E, the first day in which Employee E start working under the management of Manager M for Project P (StartDateOfIntersection). The same semantics are used to compute Attribute EndDateOfIntersection that displays the end of intersection between the time slot of E and M. Notice that the report displays the employees whose working periods intersect only with the management period of the corresponding manager, i.e., not all the employees working on Project P should be displayed. The employees records are sorted by Attribute StartDateOfIntersection. Notice that an employee can appear more than one time under a certain manager but with different date intervals.

DROP

Use the command DROP to drop all procedures and tables.

A useful strategy

Here is a useful approach for doing the project:

1. Work through the examples in Chapter 3 of “Oracle 10g Programming: A Primer, Rajshekhar Sunderrman” to become familiar with the PL/SQL syntax and creation of functions and procedures.
2. In sqlplus command mode, type “set serveroutput on size 32000”
3. Create and run simple procedures to study their behavior.
4. Always have at least two terminal windows open so that you can run sqlplus in one window and text editor on the other. Use the text editor to create .sql files to try out the functions and procedures.
5. When you build a procedure, make sure that your .sql file starts with the “CREATE OR REPLACE PROCEDURE ... ” statement, ends with the “/” to compile the procedure, and nothing else. This will make it much easier to debug your procedure. If it fails to compile, you can use the SQL command “show error” to get the detailed compilation error message that gives the line number and column number of your PL/SQL statement that has syntax problems.
6. Add the following lines to the beginning of your .sql:

rem CS 541 SQL Project 12

rem your\_first\_name your\_last\_name

rem your\_purdue\_email\_address

so that the TA can assign the grades.

(Note: Your grade may be affected if your do not follow this)

What to submit

You are going to submit ONLY ONE file for this part, p2\_your\_career\_login.sql. It should contain the following:

* @ command to run db.sql
* Function and procedure definitions
* Execute statements to execute the procedures in order
* Drop statements to clean up your database (you can just run droptables.sql)

The TA will run your submitted .sql file to grade your project.

How to submit your .sql files for Parts 1 and 2

After you are satisfied that your .sql files for both Part1 and Part2 perform the required functions, you need to create one zip file that contains both sql files for Parts1 and 2. Submit your zip file from your CS54100 account in BlackBoard before the deadline. For any questions regarding this project, please contact the TA.