CS44800 - Project 2
Spring 2017

Due: 2/13/2018 at 11:50 PM on Blackboard

Introduction

You are going to use Oracle to perform some queries against a predefined database. The schema and sample data of the database are provided. You should be able to run the project on Purdue University Linux machines. You should be able to access your password and account name in your my.cs.purdue.edu, under the “Local Accounts” section.

Remember that when you log into sqlplus your username needs to be “<username>@csora”

Information about getting your Oracle account and general initial configuration is available in: https://www.cs.purdue.edu/resources/facilities/oracle.html

A tutorial on Oracle SQL and setting up your account will be given during PSOs. Reference queries covered during these tutorials are available on the course site under this project listing as well. Feel free to contact the TAs or to attend office hours/PSO sessions if you have any questions about Oracle SQL or the project.

Your assignment

In this project you will use the file: db.sql. Copy this file into your working directory. After entering the sqlplus environment, 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. Feel free to add your own data to help with testing your queries. In order to grade the assignment, the TAs will be using a different data set for testing your queries, so make sure you do not hard-code your answers.

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(<u>UnivId</u>, UnivName)

Department(<u>DeptId</u>, DeptName)
Employee(<u>EmpId</u>, EmpName, DeptId, ZipCode)
Project(<u>ProjId,</u> ProjName)
Graduate(<u>EmpId</u>, UnivId, GradYear)
EmpProject(<u>EmpId, ProjId, StartDate</u>, EndDate)
ProjectManager(<u>ProjId, MgrId, StartDate</u>, 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 exactly 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: <u>EmpId, ProjId, StartDate</u> 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: <u>ProjId, MgrId, StartDate</u>. A NULL value in the EndDate attribute indicates "Current", i.e., the manager is currently managing that project.

Tasks

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.
Simple **SELECT** … **FROM** … **WHERE** queries should be sufficient for most queries, but some will require basic aggregation operators (e.g. COUNT(), MAX(), etc.), **GROUP BY** statements, and **ORDER BY** statements. If you are unfamiliar with Oracle SQL or any of these concepts please feel free to attend PSOs or TA office hours, or ask TAs if you have any questions.

1. Find the names of the employees who are living in West Lafayette (Zip code 47906 or 47907).
2. Find the names of the projects that are **currently** managed by any manager.
3. Display the names of all projects in descending order.

4. For each university, display the university's name and the number of employees who have graduated from that university.
5. Display the name, department name, and graduation year of each Employee.
6. Find the names of all employees who work in Department 1. Print the names in ascending order.
7. Find the names of all Employees that graduated from 'Purdue' in the year 2005 or earlier (**HINT**: Don't assume that the university ID of Purdue will be the same in the grading test cases as they are in the provided data. Use Joins to solve instead of hard-coding the university ID).
8. For each zip code in the Employee table, print the number of employees that live in that zip code. Order the results by zip code in descending order.
9. Print the name(s) of the employee(s) who graduated most recently (i.e. the highest graduation year).
10. For each entry in EmpProject, print the name of the project and the name of the employee that worked on that project. Order the results in ascending order, first by the project name and second by the employee name.

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 script to drop all tables.

What to submit

The result of your work should be a file named **p1_your_career_login.sql** which contains all the SQL statements you used in this assignment.
Please do not call @db or @drop_tables inside your submission sql script, as this complicates grading.

Add comments to your sql file with your name and purdue login, for example:

REM <name>
REM <purdue login>

An alternative comment style for SQL is to begin a line with either "--" for single-line comments or use "/*" and "*/" for block comments. **Note that "//" will not be parsed as a comment and may cause issues in grading.**

This file should be submitted on Blackboard.

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, and then call your sql script within sqlplus. Working within a text editor offers better editing functionality than the sqlplus interface available on the Purdue CS Linux machines.
5. Test your .sql file
7. Remember to divide and conquer. Test your.sql file continuously as you add the new SQL statements.