Homework #3

Complete By: Friday, July 12th @ 4:00pm
Submission: submitted digitally through Gradescope

Reading Tables

In a file named buildIndustry.sql, put the following commands.
Create a database named Industry if it does not exist. For each of the tables on the following page, drop them if they already exist, then create tables by those names with appropriate data types and cardinality restrictions. In particular, this means salary should be a number with two decimal places.

In a file named populateIndustry.sql, put the SQL necessary to populate the tables created in buildIndustry.sql with the data from the following page. This data has been included in codio in the file Industry Data.csv for your convenience, so that you can copy and paste instead of retyping the data. Instructions on how to load sql files into mysql have been included in Populate Database.txt, with the data from the University database used by the textbook given as an example. Note that the largeRelationsInsertFile.sql file takes a long time to load and should be preloaded in the database University, you may test the commands with the smallRelationsInsertFile.sql file.

Employee Table

PERSON_NAME	STREET	CITY
Sarah	Clinton	Miami
Gregory	Broadway	Chicago
Amy	State	New York
Matthew	Roosevelt	Chicago
Theresa	Main	Detroit
Omar	102nd	Louisville

Employment Table

PERSON_NAME	COMPANY_NAME	SALARY
Theresa	Facebook	15,000
Amy	Amazon	12,000
Matthew	Starbucks	9,000
Gregory	American Airline	9,000
Sarah	Amazon	14,000

Company Table

COMPANY_NAME	CITY
First Bank	Miami
Amazon	Chicago
Facebook	Detroit
Starbucks	Chicago
American Airline	Seattle
McDonalds	Milwaukee

SQL

For each of the remaining questions, put the answer in a file named Q#.sql, where # is the number of the question. For example, the answer to question 1 should be placed in a file named Q1.sql. If order is not specified, order results in alphabetical order (A-Z) for the first string attribute.

Give SQL queries that are equivalent to the following relational algebra expressions.

1. $\Pi_{PERSON\_NAME}(Employee)$

2. $\Pi_{COMPANY\_NAME}\sigma_{SALARY>10000}(Employment)$

3. $\Pi_{PERSON\_NAME,CITY}(Employment \bowtie Company)$

4. $\Pi_{PERSON\_NAME,COMPANY\_NAME}(\sigma_{CITY="Chicago"}(Employment \bowtie Employee))$

5. $\Pi_{PERSON\_NAME,COMPANY\_NAME}(\sigma_{CITY="Chicago"}(Employment \bowtie Company))$

6. Write a SQL query which returns the employee name and company name for every currently employed person in the database.

7. Write a SQL query which returns the name and salary of each person recorded in the database, but only for people whose salaries are less than 10,000, ordered from highest to lowest salary

8. Write a SQL query which returns the names of both the employees and the companies they work at, but only for employees who work at in the same city they live in. Employees live in the city indicated in the Employee table. Employees work at the city indicated in the Company table.

9. Write a SQL query which returns the names of both the employees and the companies they work at, but only for employees who work in a different city from the one they live in.

10. Write a SQL query which returns a table listing all known cities, both locations where people live as well as the cities where companies are headquartered. This should be a table with a single column.

11. Write a SQL query which lists the names of all companies along with the average salary that company pays. If a company has no known employees, the result should be NULL. Title the column containing the total salary as TOTAL_SALARY in the table your query generates.