Name: ______________________________ ID: __________

CS 480
Database Systems
Fall 2019

Homework #5

Complete By: **Monday, October 14th @ 5:00pm**

Submission: **Submitted digitally through codio - Mark as completed when done**

Reading Tables

In this assignment you will be building and working with the database from Homework 3. On the following two pages is that set of data. There are three tables extracted from the database which store between them some information the company keeping the database finds useful

To begin querying the data, you must first construct the database schema.

In the file named buildIndustry.sql, put the commands to build the Industry database (the name we're giving this set of tables), with create statements for each of the tables containing appropriate types for the attributes of those tables. In particular, this means using an appropriate date type in SQL and a text type that can exceed 255 characters for Skill_Description.

Now that you have a database, you need to fill the database with data. The tables from the assignment, while visible on the previous page, are also included in the files Employee.csv, Certified.csv, Skill.csv, and PositionRequirements.csv. Data outside the database (in the wild) can be stored in a variety of formats. These csv files contain the data in a format where each column of the table is separated by a column, and each tuple is separated by a new line. In a file named populateIndustry.sql, put the SQL necessary to populate the tables created in buildIndustry.sql with the data from the following pages.

Employee Table

Employee_ID	Employee_Fname	Employee_Lname	Employee_HireDate	Employee_Title
2345	Brian	Oates	2/14/99	DBA
3373	Franklin	Johnson	3/15/06	Purchasing Agent
4893	Patricia	Richards	6/11/08	DBA
6234	Jasmine	Patel	8/10/09	Programmer
8273	Marco	Bienz	7/28/10	Analyst
9002	Wade	Gather	5/20/14	Clerk
9283	Juan	Chavez	7/4/14	Clerk
9382	Susan	Mathis	8/2/14	Database Programmer
13383	Raymond	Matthews	3/12/16	Programmer

Certified Table

Employee_ID	Skill_ID	Certified_Date
2345	100	2/14/04
2345	110	8/9/05
2345	180	2/14/07
3373	120	6/20/13
4893	180	6/11/08
4893	220	9/20/14
6234	110	8/10/09
6234	200	8/10/09
6234	210	1/29/14
8273	110	3/8/11
8273	190	8/19/14
9002	110	5/16/15
9002	120	5/16/15
9382	140	8/2/14
9382	210	8/2/14
9382	220	5/1/15
13383	170	3/12/16

Skill Table

Skill_ID	Skill_Name	Skill_Description
100	Basic Database Management	Create and manage database user accounts
110	Basic Web Design	Create and maintain HTML and CSS documents
120	Advanced Spreadsheets	Use of advanced functions, user-defined functions, and macroing
130	Basic Process Modeling	Create core business process models using standard libraries
140	Basic Database Design	Create simple data models
150	Master Database Programming	Create integrated trigger and procedure packages for a distributed environment
160	Basic Spreadsheets	Create single tab worksheets with basic formulas
170	Basic C# Programming	Create single-tier data aware modules
180	Advanced Database Management	Manage Database Server Clusters
190	Advanced Process Modeling	Evaluate and Redesign cross-functional and external business processes
200	Advanced C# Programming	Create multi-tier applications using multi-threading
210	Basic Database Manipulation	Create simple data retrieval and manipulation statements in SQL
220	Advanced Database Manipulation	Use of advanced data manipulations methods for multi-table inserts, sets operations and correlated subqueries.

Position Requirements

Employee_Title	Skill_ID
Clerk	100
Programmer	110
Programmer	130
Programmer	170
Analyst	120
Analyst	130
Analyst	160
Analyst	140
Purchasing Agent	160
Database Programmer	140
Database Programmer	210
Database Programmer	100
Database Programmer	220
DBA	180
DBA	150

Interpreting Relational Algebra (30 points)

For each of the following three questions, put the SQL query that is equivalent to the relational algebra query in the associated Q#.sql file where # is the number of the question. That would be Q1.sql for the first problem, Q2.sql for the second problem, and so on.
More simply, convert the following Relational Algebra Queries into SQL.

1. $\Pi_{\text{Employee\_FName,Employee\_LName}}(\text{Employee})$

2. $\Pi_{\text{Employee\_ID}}\ \sigma_{\text{Employee\_HireDate>=1/1/2010}}(\text{Employee})$

3. $\Pi_{\text{Employee\_ID, Skill\_Name}}(\text{Certified} \bowtie \text{Skill})$

Writing Relational Algebra (70 points)

For each of the questions in this section, you will write a SQL query that generates the table or answer to the query.

4. Build a SQL query that will produce the following table. Keep in mind the properties of relations. (5 points)

Employee_Title
Purchasing Agent
Programmer
DBA
Database Programmer
Clerk
Analyst

5. What is each Employee's title and first name? (5 points)

6. When were certifications given for training in Basic Web Design?

7. What skill(s) is Jasmine Patel certified in? List the names of the skills.

8. Who are the employee(s) who are certified in Advanced Database Management? List the first and last names.

9. List the Titles of the employees alongside the Names of the skills for certifications given before 06/01/2009.

10. List the skills the employee with Employee_ID stored in the variable @E does not possess certification for that they should according to their title and the Position Requirements table.
For example, Brian Oates has the title of DBA, but the DBA position requires training in both Advanced Database Management and Master Database Programming, and Brian only has training in Advanced Database Management.

The output table if @E stored 2345 would be this

Skill_Name
Master Database Programming

11. Write a SQL query which constructs a table recording which employees are senior to each other, specifically for employees who are Clerks.
For example, Wade Gather is senior to Juan Chavez because Wade was hired before Juan. If a Sam Smith was hired today, with ID 11111, Juan would be senior to Sam and Wade would be senior to both Sam and Juan.
The table should contain two columns, the ID of the senior employee, and the ID of the Junior employee, for every pair of employees who are related by seniority.

For example, the table for this data set with the addition of Sam would be this

Senior_ID	**Junior_ID**
9002	**9823**
9002	**11111**
9283	**11111**