Fall B461
Assignment 1

Relational Databases, expressing queries in SQL

The goal of this assignment is to become familiar with the PostgreSQL system, to create a relational database, to examine the side-effects on the state of the database in the presence or absence of primary and secondary constraints, and to formulate some simple queries in SQL and evaluate them in PostgreSQL.

Before you can solve this assignment, you will need download PostgreSQL (version 12) and install it on your computer.

To turn in your assignment, you will need to upload to Canvas a single file with name `assignment1.sql` which contains the necessary SQL statements that solve the problems in this assignment. The `assignment1.sql` file must be such that the AI's can run it in their PostgreSQL environment. In addition, you will need to upload a separate `assignment1.txt` file that contains the results of running your queries. We have posted the exact requirements and an example for uploading your solution files. (See the module `Instructions for turning in assignments`.)

For the problems in this assignment we will the following database schema:

```
Employee(id : integer, ename:text, city:text, cname:text, salary:integer)
Company(cname : text, city : text)
JobSkill(id : integer, skill : text)
Manages(mid : integer, eid : integer)
```

In this database, we maintain a set of employees (`Employee`), a set of companies (`Company`), and a set of employee job skills (`JobSkill`). The `city`, `cname`, and `salary` attributes in `Employee` specify the city in which the employee lives, the (unique) company the employee works for, and the employee's salary at that company. The `city` attribute in `Company` indicates a city in which the company is located. (Companies may be located in multiple cities.) An employee can have multiple job skills (`JobSkill`). It is permitted that an employee has no job skills. A pair (m, e) in `Manages` indicates that m denotes an employee who is a manager of another employee denoted by e. We permit that a manager manages multiple employees and that an employee can have multiple managers. (It is possible that an employee has no manager and that an employee is not a manager.) We further require that an employee and his or her managers work for the same company. In the relation `Manages`, the attributes `mid` and `eid` refer to the `id` of the manager and the employee, respectively.

We assume the following primary key, foreign key, and referential integrity constraints:

- `id` is the primary key of `Employee`
- (`cname`, `city`) is the primary key of `Company`
- (`id`, `skill`) is the primary key of `JobSkill`
- (`mid`, `eid`) is the primary key of `Manages`
- `cname` in `Company` is the name of a company that occurs in the `Company` relation
- `id` is a foreign key in `JobSkill` referencing the primary key `id` in `Employee`
- `mid` is a foreign key in `Manages` referencing the primary key `id` in `Employee`
- `eid` is a foreign key in `Manages` referencing the primary key `id` in `Employee`

Note the files `Employee.sql`, `Company.sql`, `JobSkill.sql`, and `Manages.sql` that contain the relation instances for the `Employee`, `Company`, `JobSkill`, and `Manages` relations that are supplied for this assignment.

1 Database creation and impact of constraints on INSERT and DELETE statements.

1. Create a database in PostgreSQL that stores these relations. Make sure to specify primary and foreign keys. Then write SQL queries that return each of the relation instances `Employee`, `Company`, `JobSkill`, and `Manages`.
2. Provide 6 conceptually different examples that illustrate how the presence or absence of primary and foreign keys affect insert and deletes in these relations. To solve this problem, you will need to experiment with the `Employee`, `Company`, `JobSkill`, and `Manages` relation schemas and instances. For example, you should consider altering primary keys and foreign key constraints and then consider various sequences of insert and delete operations. You may also need to change some of the relation instances. Certain inserts and deletes should succeed but other should generate error conditions. (Consider the lecture notes about keys, foreign keys, and inserts and deletes as a guide to solve this problem.)

2 Formulating queries in SQL

For this assignment, you are required to use tuple variables in your SQL statements. You can also not use aggregate functions in this assignment. For example, in formulating the query "Find the id and name of each employee who lives in Bloomington" you should write the query

```
SELECT  e.id, e.ename
FROM    Employee e
WHERE   e.city = 'Bloomington'
```

instead of

```
SELECT  id, ename
FROM    Employee
WHERE   city = 'Bloomington'
```

Write SQL statements for the following queries. Make sure that each of your queries returns a set but not a bag. In other words, make appropriate use of the `DISTINCT` clause where necessary.

1. Find the id, name, and salary of each employee who lives in Indianapolis and whose salary is in the range $[30000, 50000]$.
2. Find the id and name of each employee who works in a city located in Chicago and who has a manager who lives in Bloomington.
3. Find the id and name of each employee who lives in the same city as at least one of his or her managers.
4. Find the id and name of each employee who has at least 3 job skills.
5. Find the id, name, and salary of each manager who manages an employee who manages at least one other employee who has a programming job skill.
6. For the pairs (id1, id2) of different employees who have a common manager.
7. Find the cname of each company that does not have employees who live in Bloomington.
8. For each company, list its name along with the ids of its employees who have the highest salary.
9. Find the id and name of each employee who does not have a manager with a salary higher than that of the employee.
10. Find the id and name of each manager who has none of the skills of the employees that he or she manages.