B561: Advanced Database Concepts
Assignment 4
Spring 2022

Due: Friday, March 11th, 11:59pm EST

This assignment covers:

- Aggregate Functions and Data Partitioning
- SQL Functions & Expressions

To turn in your assignment, you will need to upload to Canvas a single file with name assignment4.sql which contains the necessary SQL statements that solve the problems in this assignment. The assignment4.sql file must be so that the AI's can run it in their PostgreSQL environment. You should use the script file to construct the assignment4.sql file. (Note that the data to be used for this assignment is included in this file.) In addition, you will need to upload a separate assignment4.txt file that contains the results of running your queries.

Database schema and instances

For the problems in this assignment we will use the following database schema:[1]

```
Person(pid, pname, city)
Company(cname, headquarter)
Skill(skill)
worksFor(pid, cname, salary)
companyLocation(cname, city)
personSkill(pid, skill)
hasManager(eid, mid)
Knows(pid1, pid2)
```

In this database we maintain a set of persons (`Person`), a set of companies (`Company`), and a set of (job) skills (`Skill`). The `pname` attribute in `Person` is the name of the person. The `city` attribute in `Person` specifies the city in which the person lives. The `cname` attribute in `Company` is the name of the company. The `headquarter` attribute in `Company` is the name of the city wherein the company has its headquarter. The `skill` attribute in `Skill` is the name of a (job) skill. A person can work for at most one company. This information is maintained in the `worksFor` relation. (We permit that a person does not work for any company.) The `salary` attribute in `worksFor` specifies the salary made by the person. The `city` attribute in `companyLocation` indicates a city in which the company is located. (Companies may be located in multiple cities.) A person can have multiple job skills. This information is maintained in the `personSkill` relation. A job skill can be the job skill of multiple persons. (A person may not have any job skills, and

[1] The primary key, which may consist of one or more attributes, of each of these relations is underlined.

a job skill may have no persons with that skill.) A pair (e, m) in `hasManager` indicates that person e has person m as one of his or her managers. We permit that an employee has multiple managers and that a manager may manage multiple employees. (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 must work for the same company. The relation `Knows` maintains a set of pairs (p_1, p_2) where p_1 and p_2 are pids of persons. The pair (p_1, p_2) indicates that the person with pid p_1 knows the person with pid p_2. We do not assume that the relation `Knows` is symmetric: it is possible that (p_1, p_2) is in the relation but that (p_2, p_1) is not. The domain for the attributes `pid`, `pid1`, `pid2`, `salary`, `eid`, and `mid` is `integer`. The domain for all other attributes is `text`. We assume the following foreign key constraints:

- `pid` is a foreign key in `worksFor` referencing the primary key `pid` in `Person`;
- `cname` is a foreign key in `worksFor` referencing the primary key `cname` in `Company`;
- `cname` is a foreign key in `companyLocation` referencing the primary key `cname` in `Company`;
- `pid` is a foreign key in `personSkill` referencing the primary key `pid` in `Person`;
- `skill` is a foreign key in `personSkill` referencing the primary key `skill` in `Skill`;
- `eid` is a foreign key in `hasManager` referencing the primary key `pid` in `Person`;

- `mid` is a foreign key in `hasManager` referencing the primary key `pid` in `Person`;
- `pid1` is a foreign key in `Knows` referencing the primary key `pid` in `Person`; and
- `pid2` is a foreign key in `Knows` referencing the primary key `pid` in `Person`

The file `Assignment4Script.sql` contains the data supplied for this assignment.

1 Solving queries using Aggregate Functions

Formulate the following queries in SQL. **You must use aggregate functions in ALL these queries and must not use set predicates where it is mentioned explicitly**. You can use views, temporary views, parameterized views, and user-defined functions.

1. Find each pair (c, p) where **c** is the `city` and **p** is the `pid` of the person that lives in **c**, and earns the highest salary among all persons living in **c**. You must **not** use set predicates in this query.

2. Find the `pid` and `pname` of each person that knows the least amount of people (greater than 0) at the company that they work at. (The persons they know should work at the same company).You must **not** use set predicates in this query.

3. Find each pair (c, a) where **c** is the `cname` of each company that has more than one manager, and **a** is the average salary of all employees working at the company who are not managers.You must **not** use set predicates in this query.

4. Find each pair (c, n) where **c** is the `cname` of a company and **n** is the number of persons who

   (a) Work at **c** and earn strictly more than *55000*, and

   (b) Have fewer than 4 skills

   You must **not** use set predicates in this query.

5. Find the `cname` of each company, such that some person that works there knows at-least half the people that work at *Google*.

6. Find each `skill` that is the skill of a person who works at a company that pays the lowest average salary among all companies.

7. Find each triple (p_1, p_2, n), where **p1** and **p2** are `pids` of different persons, and **n** is the number of common skills between **p1** and **p2**.

8. Using the `GROUP BY` count method, define a function `personInfo` that returns for a company **c** identified by its `cname`, the triple (p, s, n), where:
   - **p** is the `pid` of a person that works at **c**
   - **s** is the `salary` of **p**
   - **n** is the number of skills **p** has

   ```
   create or replace function personInfo(c text)
   returns table (p int, s int, n int) as
   $$
   ...
   $$ language sql;
   ```

   Test this function with:

   (a) `personInfo('Apple')`

   (b) `personInfo('Amazon')`

2 Solving queries using SQL Functions & Expressions

Formulate the following queries in SQL. **You must make use of SQL functions OR expressions wherever necessary**. Furthermore, you may use supplemental user-defined functions, inbuilt PostgreSQL functions, views and parameterized views.

9. Let `Point(x int, y int)` be a binary relation. Each pair (x, y) in `Point` represents a point in 2-D Space.
   With the data given in the script file, write a SQL query that generates a tuple (x_1, y_1, x_2, y_2) of different points (x_1, y_1) and (x_2, y_2), such that:

   (a) $x_1 \neq x_2$ and $y_1 \neq y_2$, and

   (b) $x_1 + y_1 = x_2 + y_2$, and

   (c) $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} > 2$

10. In statistics, the *standard deviation* is a measure of the amount of variation or dispersion of a set of values. A low standard deviation indicates that the values tend to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the values are spread out over a wider range. You can read more about it here:
    `https://en.wikipedia.org/wiki/Standard_deviation`

Define a function `SalaryStandardDeviation(cname text)`, that returns the standard deviation of the salaries of the people working at the company identified by `cname`.

```
create or replace function SalaryStandardDeviation(cname
text)
returns table (std_deviation float) as
$$
...
$$ language sql;
```

Some inbuilt functions that may come in handy:
https://www.postgresql.org/docs/7.4/functions-math.html

Test this function with:

(a) `SalaryStandardDeviation('Apple')`

(b) `SalaryStandardDeviation('Amazon')`

(c) `SalaryStandardDeviation('Google')`

(d) `SalaryStandardDeviation('Netflix')`