Fall 2024 B561 Assignment 3

Chenghong Wang

October 23, 2024

This assignment relies on the lectures:

- Functions and expressions in SQL;
- Aggregate functions and partitioning;
- Triggers; and
- Queries with quantifiers.

To turn in your assignment, you will need to upload to Canvas a single file with name `assignment3.sql` which contains the necessary SQL statements that solve the problems in this assignment. The `assignment3.sql` file must be so that the AI's can run it in their PostgreSQL environment. You should use the `Assignment-Script-2023-Spring-assignment3.sql` file to construct the `assignment3.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 `assignment3.txt` file that contains the results of running your queries and `assignment3.pdf` file that contains the venn diagrams with conditions.

1 Database schema and instances

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

```
Student(sid, sname, city)
Company(cname, headquarter)
Skill(skill)
worksFor(sid, cname, salary)
companyLocation(cname, city)
StudentSkill(sid, skill)
hasManager(eid, mid)
Knows(sid1, sid2)
```

In this database we maintain a set of Students (`Student`), a set of companies (`Company`), and a set of (job) skills (`Skill`). The `sname` attribute in `Student` is the name of the Student. The `city` attribute in `Student` specifies the city in which the Student 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 Student can work for at most one company. This information is maintained in the `worksFor` relation. (We permit that a Student does not work for any company.) The `salary` attribute in `worksFor` specifies the salary made by the Student.

The `city` attribute in `companyLocation` indicates a city in which the company is located. (Companies may be located in multiple cities.)

A Student can have multiple job skills. This information is maintained in the `StudentSkill` relation. A job skill can be the job skill of multiple Students. (A Student may not have any job skills, and a job skill may have no Students with that skill.)

A pair (`e`;`m`) in `hasManager` indicates that Student `e` has Student `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 (`p1`; `p2`) where `p1` and `p2` are `sids` of Students. The pair (`p1`; `p2`) indicates that the Student with `sid p1` knows the Student with `sid p2`. We do not assume that the relation `Knows` is symmetric: it is possible that (`p1`; `p2`) is in the relation but that (`p2`; `p1`) is not.

The domain for the attributes `sid`, `sid1`, `sid2`, `salary`, `eid`, and `mid` is integer. The domain for all other attributes is text.

We assume the following foreign key constraints:

- `sid` is a foreign key in `worksFor` referencing the primary key `sid` in `Student`;
- `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`;
- `sid` is a foreign key in `StudentSkill` referencing the primary key `sid` in `Student`;
- `skill` is a foreign key in `StudentSkill` referencing the primary key `skill` in `Skill`;
- `eid` is a foreign key in `hasManager` referencing the primary key `sid` in `Student`;
- `mid` is a foreign key in `hasManager` referencing the primary key `sid` in `Student`;
- `sid1` is a foreign key in `Knows` referencing the primary key `sid` in `Student`;
- `sid2` is a foreign key in `Knows` referencing the primary key `sid` in `Student`.

The file assignment3Script.sql contains the data supplied for this assignment.

2 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 sid of the Student that lives in c, and earns the lowest salary among all Students living in c. You must not use set predicates in this query.

2. Find the sid and name of each Student who has fewer than 2 of the combined set of job skills of Students who work for Netflix. By combined set of jobskills we mean the set

$$\{s \mid s \textit{ is a jobskill of an employee of } Netflix\ \}$$

3. Find each pairs (s1; s2) of skills such that the set of Students with skill s1 is the same as the set of Students with skill s2.

4. Find each sid of a Student who knows at least two people who (a) work for Apple and (b) who make less than 55000.You must not use set predicates in this query.

5. Find the cname of each company, such that some Student that works there knows at-least quarter of the people that work at Amazon.You must not use set predicates in this query.

6. Find each pair (c, a) where c is the cname of each company that has at least 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.

7. (a) Using the GROUP BY count method, define a function

```
create or replace function numberOfSkills(c text)
returns table (sid integer, salary int, numberOfSkills bigint) as
$$
...
$$ language sql;
```

   that returns for a company identified by its cname, each triple (p, s, n) where (1) p is the sid of a Student who is employed by that company, (2) s is the salary of p, and (3) n is the number of job skills of p. (Note that a Student may not have any job skills.)

   (b) Test your function for Problem 7a for the companies Apple, Amazon, and ACM.

(c) Write the same function numberOfSkills as in Problem 7a but this time without using the GROUP BY clause.

(d) Test your function for Problem 7c for the companies Apple, Amazon, and ACM.

(e) Using the function numberOfSkills but without using set predicates, write the following query: "Find each pair (c; p) where c is the name of a company and where p is the sid of a Student who (1) works for company c, (2) makes more than 50000 and (3) has the most job skills among all the employees who work for company c."

3 Queries with quantifiers

Using the method of Venn diagrams with conditions (Show these venn diagrams with conditions in pdf file), write SQL queries for the following queries with quantifiers.
To get full credit in these problems, you must write appropriate views and parameterized views for the sets A and B that occur in the Venn diagram with conditions (Show these venn diagrams with conditions in pdf file) for these queries. (See the lecture on Queries with Quantifiers.)
Hint: You can create views, functions and then use them in your query to find the answer.
Make the following two queries without using the COUNT function:

8. Find the sid and name of each Student who knows all the Students who (a) live in Bloomington, (b) make at least 55000, and (c) have at least one skill.

9. Find the cname of each company who only employs man- agers who make more than 50000.

Make the following query using the COUNT function: (Show the venn diagrams with conditions in pdf file)

10. Find the sid and name of each Student who knows at least 3 people who each have at most 2 managers.

11. Find the cname of each company that employs an even number of Students who have at least 2 skills.

12. Find the pairs (p1, p2) of different Student sids such that the Student with sid p1 and the Student with sid p2 have the same number of skills.

4 Triggers

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

13. Explain how triggers can be used to implement the Primary key Constraint, with an example.(You are not allowed to use postgres cascade)

14. Explain how triggers can be used to implement the Refer- ential Integrity Constraint, with an example.(You are not allowed to use postgres cascade)

15. Consider two relations R(A:integer,B:integer) and S(B:integer) and a view with the following definition:

```
select distinct r.A
from R r, S s
where r.A > 10 and r.B = s.B;
```

    Suppose we want to maintain this view as a materialized view called V(A:integer) upon the insertion of tuples in R and in S. (You do not have to consider deletions in this question.)
    Define SQL insert triggers and their associated trigger func- tions on the relations R and S that implement this mate- rialized view. Write your trigger functions in the language 'plpgsql.'
    Make sure that your trigger functions act in an incremen- tal way and that no duplicates appear in the materialized view.