B461 Database Concepts
Assignment 6
Fall 2023

This assignment tests the following concepts:

- Nested Relations and Semi-Structured Databases
- PL/pgSQL

To turn in your assignment, you will need to upload to Canvas a single file with name assignment6.sql which contains the necessary SQL statements that solve the problems in this assignment. The assignment6.sql file must be so that the AI's can run it in their PostgreSQL environment. You should use the assignment6Script.sql file to construct the assignment6.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 assignment6.txt file that contains the results of running your queries.

1 Nested Relations and Semi-structured databases

Consider the lecture on Nested relational and semi-structured databases. In that lecture, we considered the `studentGrades` nested relation and the `jstudentGrades` semi-structured database, and we constructed these using a PostgreSQL query starting from the Enroll relation.

1. Write a PostgreSQL view `courseGrades` that creates the nested relation of type $(\text{cno}, \text{gradeInfo}\{(\text{grade}, \text{students}\{(\text{sid})\})\})$ This view should compute for each course, the grade information of the students enrolled in this course. In particular, for each course and for each grade, this relation stores in a set the students who obtained that grade in that course. [**15 points**]

2. Starting from the `courseGrades` view in Problem 1, solve the following query: Find each (s, C) pair where s is the sid of a student and C is the set of cnos of courses in which the student received an 'A' or a 'B' but not a 'C'. The type of your answer relation should be $(\text{sid} : \text{text}, \text{Courses} : \{(\text{cno} : \text{text})\})$. [**15 points**]

3. Write a PostgreSQL view `jcourseGrades` that creates a semi-structured relation which stores jsonb objects whose structure conforms with the structure of tuples as described for the `courseGrades` in Problem 8. Test your view. [**15 points**]

2 Object Relational Programming

The following problems require you to write object-relational programs. Many of these require programs written in Postgres' PL/pgSQL database programming language.

4. Write a PL/pgSQL function that takes an array of integers as input and returns a new array where each element is the sum of all the elements in the input array up to and including that element. For example, if the input array is $\{1, 2, 3, 4\}$, the output array should be $\{1, 3, 6, 10\}$. [**15 points**]

5. Write a function that takes a positive integer as input and returns an array of all the prime numbers up to and including that integer. [**15 points**]

6. Consider a parent-child relation PC(parent, child). (You can assume that PC is a rooted tree and the domain of the attributes parent and child is int.) An edge (p, c) in PC indicates that node p is a parent of node c. Now consider a pair of nodes (m, n) in PC (m and n may be the same nodes.) We say that m and n are in the same generation when the distance from m to the root of PC is the same as the distance from n to the root of PC. Consider the following recursive query that computes the `sameGeneration` relation:

```
WITH RECURSIVE sameGeneration(m, n) AS
((SELECT parent, parent FROM PC)
UNION
(select child, child from PC)
UNION
SELECT t1.child, t2.child
FROM sameGeneration pair, PC t1, PC t2
WHERE pair.m = t1.parent and pair.n = t2.parent)
select distinct pair.m, pair.n from sameGeneration pair order by m, n;
```

Write a non-recursive function `sameGeneration()` in the language PL/pgSQL that computes the `sameGeneration` relation.

[25 points]