Studio: Let's talk about sets!

Introduction

In this studio you will:

Get practice writing queries with JOIN

Explore using relational algebra to describe queries

Constructing query trees

Examine JOIN performance

You are encouraged to work in groups of up to 4 people. Please do not just let one person do all of the work while everyone else watches. It is important for everyone to follow these steps and participate in the studio.

Before you begin

We will be using the course tracking database that you designed for this studio. If you have not finished adding the relationships (foreign keys) to your database, please do that before proceeding.

It will also be useful for you to have some sample data to use. Please take a moment to make sure that each table has a few rows of data available so that you can check your work.

Query Practice

Use your course tracking database to answer the below questions. Please note that your answers may vary depending on the design of your database. Do the best that you can!

* Write a query that well retrieve a list of all students currently enrolled in the school. This list should only contain their first and last names. Determine the relational algebra operations that must be performed to satisfy this query, and construct a query tree.

* Write a query that will show which courses are offered by which departments. If you do not have a Department table, you

should incorporate it into your design and add it to your database, making sure to include the appropriate foreign keys. The result of this query should have one row for each course that contains the name of the department and the name of the course offered by the department. Determine the relational algebra operations that must be performed to satisfy this query, and construct a query tree.

* Select one course that exists in your database, and write a query that will show all of the students who have taken or are currently taking that course. This will likely require a join of two or more tables. Determine the relational algebra operations that must be performed to satisfy this query, and construct a query tree.

* Select one student that exists in your database, and write a query that will show what courses they have taken. Determine the relational algebra operations that must be performed to satisfy this query, and construct a query tree.

* You are given the following relational algebra expression:

$$\pi_{sid}(\bowtie_{enrollments.sid=students.sid}(\bowtie_{course.courseno=enrollments.courseno}(\bowtie_{department.depno=course.depno}(\sigma_{departmentname=computerscience}(Departments), Courses), Enrollments), Students))$$

Translate this expression into a query tree, and then translate it inst a SQL expression.

* Write a query that lists the number of students that are enrolled in each course. Determine the relational algebra operations that must be performed to satisfy this query, and construct a query tree.

* Rewrite each of the above queries using tuple calculus

Join Performance

In class we discussed how joins work at a high level. The performance of joins is something of particular interest to us and something we will talk about in more detail soon. For now, it is worthwhile to explore what things will affect join performance.

Using the employees database, write queries to answer the following questions. All of these will require a join. For each query, pay attention to how long the query takes.

* List the first name, last name, and all salaries (including historical salaries) for each employee.

* List the first name, last name, and current salary (not including historical salaries) for each employee.

* List the first name, last name, current salary, and current title for each employee.

* List the first and last name of each employee along with the first and last name of their manager.

Based on the above queries, do your best to answer the following questions. It is okay if you cannot provide specific answers, we will discuss this in much more depth in the coming weeks.

* How is join performance affected by the number of rows in the tables being joined?

* How is join performance affected by filtering (WHERE)?

* How is join performance affected when multiple tables (more than 2) need to be joined?