Database Systems, CSCI 4380-01
Homework # 1 Answers
Due Friday September 14, 2018 at 11:59:59 PM

Homework Statement. This homework is worth 4% of your total grade. If you choose to skip it, Midterm #1 will be worth 4% more. Remember, practice is extremely important to do well in this class. I recommend that not only you solve this homework, but also work on homeworks from past semesters. Link to those is provided in the Piazza resources page.

This homework aims to teach you how to construct complex queries using relational algebra. Please do the parts in sequence. The questions get harder and build on your knowledge of relational algebra from previous parts. Each question is equal weight.

Database Description. Suppose you are given the following database for keeping track of grades in this course.

```
students(rin, fname, lname, email, optin_date, optout_date)

exams(name, id, exam_date, maxgrade, points)

examgrades(rin, name, id, grade)

hws(id, name, points, given_date, due_date, maxgrade, nextexam_name, nextexam_id)

hwgrades(rin, id, submission_date, grade)
```

Each `student` may have an opt-in date, if there is no date (i.e. the value is NULL) then the homeworks are optional for this student. If there is an opt-in and no opt-out date, then homeworks are required. If there are dates for both opt-in and opt-out, only the homeworks that have a due date within within the given dates are required.

The `exams` relation contains both quizzes and exams, stored in `name, id` fields as `'Exam', 1, 'Exam', 2, 'Exam', 3` (for final), or `'Quiz', 1, 'Quiz', 2`, etc. The id is used to show which exam comes after another.

The grade for each exam references both the name and the id of the exam for each student and their grade.

Homeworks are stored similarly in the `hws` relation with id 1,2,3, etc. For each homework, the upcoming (midterm or final) exam is listed. As discussed, points for skipped homeworks will count towards the next exam. The `name` for each homework is `'hw'` for simplicity.

The grades for each homework is stored similarly in `hwgrades` where `id` is the id of the homework.

Note: All date fields are formatted as `mon-day-year`, e.g. `01-31-2016`. You can assume that you can check if a date value `X` comes after another value `Y` by checking whether `X > Y`.

Write the following queries using relational algebra (pay attention to the attributes required in the output!):

Question 1. The following queries only need a single SELECT (σ), followed by a PROJECT (π) and RENAMING (ρ) as necessary:

(a) Return the first and last name of all students who have opted in to the homeworks before `'9/25/2018'` and did not opt out.

Answer.

$$\Pi_{fname,lname}(\sigma_{\text{optin\_date < '9/25/2018' and optout\_date=NULL}}\ STUDENTS)$$

NOTE: In SQL, you would use `is NULL` instead of `= NULL`. For relational algebra, either one is fine.

(b) Return the id and points for all homeworks that are due before `'Exam', 2`.

Answer.

$$\Pi_{id,points}(\sigma_{\text{nextexam\_name='Exam' and nextexam\_id=2}}\ HWS)$$

Question 2. The following queries combine SELECT (σ), SET operations ($\cap, \cup, -$), PROJECTION (π) and RENAMING (ρ) as necessary:

(a) Find and return the name, id and points of all exams and homeworks given or had due dates before ’11/22/2018’.

Answer.

$$\Pi_{id,name,points}(\sigma_{\text{due\_date} \,<\, \text{'11/22/2018'}}\ EXAMS) \cup \Pi_{id,name,points}(\sigma_{\text{exam\_date} \,<\, \text{'11/22/2018'}}\ HWS)$$

(b) Find the id of homeworks that no student has submitted.

Answer.

$$(\Pi_{id}\ HWS) - (\Pi_{id}\ HWGRADES)$$

(c) Find and return the RIN of all students who turned in both homeworks 1 and 2.

Answer.

$$(\Pi_{rin}(\sigma_{\text{id=1}}\ HWGRADES)) \cap (\Pi_{rin}(\sigma_{\text{id=2}}\ HWGRADES))$$

Obviously, this can also be done with a join, join hwgrades with itself on same RIN. Try this yourself as an exercise. Since we require here a set operation, we show the set intersection method.

Question 3. The following queries combine SELECT (σ) statements with any number of JOINS as neede ($\bowtie$, theta or natural) (or CARTESIAN PRODUCT), followed by a PROJECT (π) and RENAMING (ρ) as necessary:

(a) Find the first and last name of all students who submitted homework #1.

Answer.

$$\Pi_{fname,lname}(\sigma_{\text{id=1}}\ HWGRADES) \bowtie STUDENTS)$$

Note that given hwgrades and students have only one attribute in common, RIN, the above natural join simply finds tuples with the same RIN value. You can also use regular (theta) join, in which case, you must first rename the rin attribute in one of the relations first.

(b) Find the RIN and Exam 1 grades of all students who submitted at least one homework due before Exam 1.

Answer.

$$\begin{aligned} R1 &= \Pi_{rin}(\sigma_{\text{nextexam\_name='Exam' and nextexam\_id = 1}}\ HWS \bowtie HWGRADES) \\ R2 &= \Pi_{rin,grade}(R1 \bowtie (\sigma_{\text{name='Exam' and id=1}}\ EXAMGRADES) \end{aligned}$$

Question 4. Freeform, you decide which combination is needed. Any relational algebra operator is fine. Remember to construct these in parts and provide comments on what each part is computing. This will make it possible for us to give partial credit.

(a) For each student, return their RIN, name, exam/homework name, exam/homework id, grade and points for all exams (including quizzes) and homeworks that they have a grade for.

Answer.

$$\begin{aligned} R1 &= \Pi_{rin,id,name,grade,points}(\sigma_{\text{grade <> NULL}}\ EXAMGRADES \bowtie EXAMS) \\ R2 &= \Pi_{rin,id,name,grade,points}(\sigma_{\text{grade <> NULL}}\ HWGRADES \bowtie HWS) \\ Result &= R1 \cup R2 \end{aligned}$$

The natural join for EXAMGRADES and EXAMS is over id and name. The natural join for HWGRADES and HWS is over id. As we were not specific about whether grade can be empty or not, you can also skip the grade not null condition. Note that <> is the SQL version of not equal to. We are putting it here to get you used to it.

(b) Find and return the RIN and Exam 1 grade of the students who did not complete any of the homeworks before Exam 1.

Answer.

$$\begin{aligned} R0 &= \sigma_{\text{grade <> NULL}}\ HWGRADES \\ R1 &= \Pi_{rin}(\sigma_{\text{nextexam\_name='Exam' and nextexam\_id=1}}\ HWS \bowtie R0) \\ R2 &= (\Pi_{rin}\ STUDENTS) - R1 \\ Result &= \Pi_{rin,grade}(R2 \bowtie (\sigma_{\text{name='Exam' and id=1}}\ EXAMS)) \end{aligned}$$

R1 is all students who completed at least one homework before Exam 1. R2 is students who did not complete any homework before Exam 1.

Extra challenge questions.

If you are finished with all these queries but find yourself in need of a personal challenge, try to write this query to explore the expressive power of relational algebra (no hw credit for these questions):

Extra Question 1: Try to find students who completed two homeworks in a row.

Answer.

This is similar to counting to 2. We will find for each student, two tuples in the hwsubmissions for two homeworks in a row.

$$\begin{aligned} hwg2(rin2, id2, grade2) &= \Pi_{rin,id,grade}\, HWGRADES \\ R2 &= \Pi_{rin}(HWGRADES \bowtie_{\text{rin=rin2 and id2=id+1}} hwg2) \\ Result &= \Pi_{lname,fname}(R2 \bowtie STUDENTS) \end{aligned}$$

The last join is needed if the question asks the first and last name (not specified here explicitly).

Extra Question 2: Find and return the RIN of all students who have completed all the assigned homeworks.

Answer. This is a more tricky question. In essence, you will do a double negation: find all students for whom there is no homework they did not complete.

$$\begin{aligned} R1 &= (\Pi_{id}\, HWS) \times (\Pi_{rin}\, STUDENTS) \\ R2 &= \Pi_{id,rin}\, (\sigma_{\text{grade <> NULL}}\, HWGRADES) \\ R3 &= R1 - R2 \\ R4 &= \Pi_{rin}(R3) \\ R5 &= (\Pi_{rin}\, HWGRADES) - R4 \end{aligned}$$

R1 is the relation that contains tuples for cases where a student completed all the homeworks, i.e. all tuples that would exist for any student if they completed all the homeworks. R3 subtracts all homeworks completed. A student will have a RIN in R4 if they had a homework that they did not complete. So, what we want is students that are not in R4. That is what R5 finds. Note that we use HWGRADES, not STUDENTS here because we want to make sure that we do not return students who did not turn in any homework.

Let's see an example of how this works:

id	rin	id	rin	id	rin	id
1	a	1	a	1	a	2
2	b	1	a	2		
	b	2	b	1		
			b	2		
HWS	HWGRADES		R1		R3	

As you can see in the above example, b has done both homeworks while a missed one. So in R3, we have tuple for a but none for b. So, we must return b by subtracting students in R3 from all students who submitted a homework (as shown in R5).