University of Toronto
CSC343, Fall 2017

Relational Algebra Exercises

These are solutions to some of the exercises we worked on in class. The remaining solutions will be posted once we have finished the exercises. **Important:** There are other good answers to each of these queries.

Schema

Note: "breadth" is a boolean indicating whether or not a course satisfies the breadth requirement for degrees in the Faculty of Arts and Science.

Student(sID, surName, firstName, campus, email, cgpa)

Course(dept, cNum, name, breadth)

Offering(oID, dept, cNum, term, instructor)

Took(sID, oID, grade)

Offering[dept, cNum] $\subseteq$ Course[dept, cNum]

Took[sID] $\subseteq$ Student[sID]

Took[oID] $\subseteq$ Offering[oID]

Queries

Write a query for each of the following:

1. Student number of all students who have taken csc343.

   Answer:

   $\Pi_{sID}\sigma_{dept=\text{“}csc\text{”}\wedge cNum=343}(Took \bowtie Offering)$

2. Student number of all students who have taken csc343 and earned an A+ in it.

   Answer:

   $Good343(sID) := \Pi_{sID}\sigma_{dept=\text{“}csc\text{”}\wedge cNum=343\wedge grade\geq 90}(Took \bowtie Offering)$

3. The names of all such students.

   Answer:

   Here we reuse relation Good343 from the previous question.

   $\Pi_{surName,firstName}(Good343 \bowtie Student)$

try to compose solutions

4. The names of all students who have passed a breadth course with Professor Picky.

   Answer:

   $PickyBreadth(oID) := \Pi_{oID}\sigma_{breadth=true \wedge instructor=\text{“}Picky\text{”}}(Course \bowtie Offering)$

   $Passers(sID) := \Pi_{sID}\sigma_{grade \geq 50}(PickyBreadth \bowtie Took)$

   $Answer(surName, firstName) := \Pi_{surName,firstName}(Passers \bowtie Student)$

cannot infer this by looking at one row.
So think of set intersection, or self-joining

5. sID of all students who have earned some grade over 80 and some grade below 50.

   Answer:

   $(\Pi_{sID}\sigma_{grade>80}Took) \cap (\Pi_{sID}\sigma_{grade<50}Took)$

6. Terms when Cook and Pitassi were both teaching something.

   Answer:

   $(\Pi_{term}\sigma_{instructor=\text{“}Cook\text{”}}\ Offering) \cap (\Pi_{term}\sigma_{instructor=\text{“}Pitassi\text{”}}\ Offering)$

7. Terms when either of them was teaching csc463.

   Answer:

equivalent to set union? yes

   $\Pi_{term}(\sigma_{dept=\text{“}csc\text{”} \wedge cNum=463 \wedge (instructor=\text{“}Cook\text{”} \vee instructor=\text{“}Pitassi\text{”})}\ Offering)$

8. sID of students who have earned a grade of 85 or more, or who have passed a course taught by Atwood.

seeing or, think of set union

   Answer:

   $HaveHighGrade(sID) = \Pi_{sID}\sigma_{grade \geq 85}Took$

   $PassedAtwood(sID) = \Pi_{sID}\sigma_{instructor=\text{“}Atwood\text{”} \wedge grade \geq 50}(Took \bowtie Offering)$

   $Answer(sID) := HaveHighGrade \cup PassedAtwood$

9. Terms when csc369 was not offered.

only know if csc369 is offered in a row
has to use set difference

   Answer:

   $(\Pi_{term}\ Offering) - (\Pi_{term}\sigma_{dept=\text{“}csc\text{”} \wedge cNum=369}\ Offering)$

10. Department and course number of courses that have never been offered.

    Answer:

know that a course is offered,
so use all available courses set minus ones that is offered

$(\Pi_{dept,cNum}\, Course) - (\Pi_{dept,cNum}\, Offering)$

think of self-joining

11. SIDs and surnames of all pairs of students who've taken a course together.

Answer:

$Pairs(sID1, sID2) := \Pi_{T1.sID,T2.sID}\sigma_{T1.sID<T2.sID \wedge T1.oID=T2.oID}[(\rho_{T1}Took) \times (\rho_{T2}Took)]$

$OneName(sID1, sID2, name1) := \Pi_{sID1,sID2,surName}\sigma_{sID1=sID}(Pairs \times Student)$

$Answer(sID1, sID2, name1, name2) :=$

$\Pi_{sID1,sID2,name1,surName}\sigma_{sID2=sID}(OneName \times Student)$

The strategy used here is to first identify the entities that should be in the answer by their keys, and then later add in the other required attributes. If many additional attributes are required in the answer, this is cleaner. Because you end up working with fewer attributes through the hard parts of the query, it also avoids accidentally doing an inappropriate natural join – one that would force attributes to match that we don't actually need to match. Here, I brought in one surname at a time. Rather than getting the surnames in two steps we could have done it all at once. To do so, we'd have to bring in two copies of Student, and therefore would need to do some renaming.

When there aren't many other attributes needed in the final answer, it may be simpler to keep them all along. Here's that solution:

– Who took what, and their surname.

$TookName(sID, oID, name) :=$

$\Pi_{sID,oID,surName}(Took \bowtie Student)$

$Answer(sID1, sID2, name1, name2) :=$

$\Pi_{T1.sID,T2.sID,T1.name,T2.name}\sigma_{T1.sID<T2.sID \wedge T1.oID=T2.oID}[(\rho_{T1}TookName) \times (\rho_{T2}TookName)]$

subtract from all student, students that is not the highest, which we can verify during selection

12. sID of student(s) with the highest grade in csc343, in term 20099.

tricky

Answer:

This works except for one wrinkle discussed below:

$Takers(sID, grade) := \Pi_{sID,grade}[(\sigma_{dept="csc" \wedge cNum=343 \wedge term=20099}\, Offering) \bowtie Took]$

$NotTop(sID) := \Pi_{T1.sID}\sigma_{T1.grade<T2.grade}[(\rho_{T1}Takers) \times (\rho_{T2}Takers)]$

$Answer(sID) := (\Pi_{sID}Takers) - NotTop$

Notice that the schema doesn't disallow having two different offerings of csc343 in 20099, and this makes sense in our domain. For instance, we have a day and an evening section of csc343 this term. The schema also does not prevent a student from being in the Took relation for both of them, even though this probably doesn't make sense in our domain. If such a student were to get the top mark for one and a non-top mark for the other, they would appear in NotTop and therefore not in Answer. To prevent that, we can keep the oID until the very end:

$Takers(sID, oID, grade) := \Pi_{sID,oID,grade}(\sigma_{dept="csc" \wedge cNum=343 \wedge term=20099}\, Offering) \bowtie Took$

$NotTop(sID, oID, grade) := \Pi_{T1.sID,T1.oID,T1.grade}\sigma_{T1.grade<T2.grade}[(\rho_{T1}Takers) \times (\rho_{T2}Takers)]$

$Answer(sID) := \Pi_{sID}(Takers - NotTop)$

Notice that the final projection onto sID has moved. Make sure you know why. These small matters

often make the difference between a correct and an incorrect or even syntically ill-formed query.

13. sID of students who have a grade of 100 at least twice.

    Answer:

    $$AtLeastTwice(sID) :=$$
    $$\Pi_{T1.sID}\sigma_{T1.oID\neq T2.oID\wedge T1.sID=T2.sID\wedge T1.grade=100\wedge T2.grade=100}[(\rho_{T1}Took)\times(\rho_{T2}Took)]$$

    same person, difference offering, both grade = 100

14. sID of students who have a grade of 100 exactly twice.

    Answer:

    $$AtLeastThrice(sID) :=$$
    $$\Pi_{T1.sID}$$
    $$\sigma_{T1.oID<T2.oID<T3.oID\wedge T1.sID=T2.sID=T3.sID\wedge T1.grade=T2.grade=T3.grade=100}$$
    $$[(\rho_{T1}Took)\times(\rho_{T2}Took)\times(\rho_{T3}Took)]$$
    $$ExactlyTwice(sID) := atLeastTwice - AtLeastThrice$$

    Notice that, since $\neq$ is not transitive, we can't string together $T1.oID \neq T2.oID \neq T3.oID$ and still enforce that $T1.oID \neq T3.oID$ We could compare the three oIDs all three ways, but this use of "<" works too and is more concise.

15. sID of students who have a grade of 100 at most twice.

    Answer:

    $$(\Pi_{sID}Student) - AtLeastThrice$$

16. Department and cNum of all courses that have been taught in every term when csc448 was taught.

    Answer: ShouldHaveBean: all possible combination of courses, term combination. idea is that if a course did indeed taught in every term when csc448 were taught is should overlap completely with ShouldHaveBeen partly

    $$448Terms(term) := \Pi_{term}(\sigma_{dept=\text{“}csc\text{”}\wedge cNum=448}Offering)$$
    $$CourseTerms(dept, cNum, term) := \Pi_{dept,cNum,term}Offering$$
    $$ShouldHaveBeen(dept, cNum, term) := \Pi_{dept,cNum}CourseTerms \times 448Terms$$
    $$WereNotAlways(dept, cNum, term) := ShouldHaveBeen - CourseTerms$$
    $$Answer(dept, cNum) := (\Pi_{dept,cNum}CourseTerms) - (\Pi_{dept,cNum}WereNotAlways)$$

17. Name of all students who have taken, at some point, every course Gries has taught (but not necessarily taken them from Gries).

    Answer:

    This one is very similar to question 16. Try it on your own. It is great practise for the "every" strategy.

Integrity Constraints

Use the notation

<relational algebra expression> = $\emptyset$

to write an integrity constraint for each of the following.

1. Courses at the 400-level cannot count for breadth.

   Answer:

   $\sigma_{400 \leq cNum < 500 \wedge breadth} Course = \emptyset$

2. CSC490 can only be offered at the same time as CSC454.

   Answer:

   $490Terms(term) := \Pi_{term}(\sigma_{dept=`CSC' \wedge cNum=490} Offering)$

   $454Terms(term) := \Pi_{term}(\sigma_{dept=`CSC' \wedge cNum=454} Offering)$

   $490Terms - 454Terms = \emptyset$