B561 Assignment 4: Relational Algebra

This assignment is designed to test your knowledge of the following lectures:

- Lecture 10: Relational Algebra
- Lecture 11: (Regular) Joins and semijoins
- Lecture 12: Set joins and Set semijoins in RA

For this assignment, you will need to submit 3 files. The first file is a .sql file that should contain all the SQL code relating to problems requesting the development of such code. The second file is a .txt file that should contain the output of the queries. The third file is a .pdf file that should contain your solutions for problems where RA expressions in their standard (i.e., non SQL) notation are requested. Ideally you should use latex to construct this .pdf file. Latex offers a convenient syntax to formulate RA expessions.

1 Formulating Queries in RA

1.1 Discussion

Before you solve the problems in this section, we briefly review how you can express RA expressions in SQL in a way that closely mimic their specifications as RA expressions in standard notation[1].

Consider a relation $R(A, B)$ and a relation $S(B, C)$ and consider the following RA expression F:

$$\pi_A(R) - \pi_A(\sigma_{B=1}(R \bowtie_{R.B=S.C} S))$$

Then we can write this query in SQL in a variety of ways that closely mimic its RA formulation. One way to write this RA expression in SQL is as follows:

```
SELECT DISTINCT A
FROM   R
EXCEPT
SELECT A
FROM   (SELECT DISTINCT A, B, C
        FROM   R JOIN S ON (R.B = S.C)
        WHERE  A = 1) q
```

An alternative way to write this query is to use the `WITH` statement of SQL.[2] To do this, we separate the RA expression F into sub-expressions as follows. (In this example, notice that each sub-expression corresponds to the application of a single RA operation. More generally, one can of course use sub-expressions that can contain multiple RA operations.)

[1] By RA expressions in standard notation, we refer to expressions that use the notations $(A : \mathbf{a})$, $\sigma_{\ldots}(\cdot)$, $\pi_{\ldots}(\cdot)$, $\cup$, $\cap$, $-$, $\times$, $\bowtie$, $\bowtie_{\ldots}$, $\ltimes$, and $\overline{\ltimes}$ for the RA operations. (For more detail, consult the lectures relating to RA and joins.)

[2] This is especially convenient when the RA expression is long and complicated.

Expression Name	RA expression
E_1	$\pi_A(R)$
E_2	$R \bowtie_{R.B=S.C} S$
E_3	$\sigma_{B=1}(E_2)$
E_4	$\pi_A(E_3)$
F	$E_1 - E_4$

Then we can write the following SQL query. Notice how the expressions $E1$, $E2$, $E3$, and $E4$ occur as separate queries (temporary views) in the WITH statement and that the final query gives the result for the expression F.[3]

```
WITH
E1 AS (SELECT DISTINCT A FROM R),
E2 AS (SELECT DISTINCT A, B, C FROM (R JOIN S ON (R.B = S.C)) e2),
E3 AS (SELECT A, B, C FROM E2 WHERE B = 1),
E4 AS (SELECT DISTINCT A FROM E3)
(SELECT A FROM E1) EXCEPT (SELECT A FROM E4);
```

In your answer to a problem, you may write the resulting RA expression with or without the WITH statement. (Your SQL query should of course closely resemble the RA expression it is aimed to express.). It should also be clear that in your solutions, you can not use the SQL set predicates [`NOT`] `EXISTS`, θ `ALL`, θ `SOME`, and [`NOT`] `IN`. You can also not use `GROUP BY` and aggregate functions.

[3]For better readability, I have used relational-name overloading. Sometimes, you may need to introduce new attribute names in SELECT clauses using the AS clause. Also, use DISTINCT were needed.

1.2 Problems

In the following problems, we will use the database schema that was used in Assignment 2 and Assignment3. To test your queries, you can use the `data.sql` file provided for this assignment.

Write the following queries as RA expressions in the standard RA notation.[4] In the expressions, avoid using the $\times$ operator. Rather, the use of the natural join $\bowtie$, the join $\bowtie_{\ldots}$, the semijoin $\ltimes$, and the anti semijoin $\overline{\ltimes}$ operation is encouraged. It is also not required that you "optimize" the expressions. It is sufficient that they are correct.

When you formulate your RA expressions, you can use the following abbreviations for the relations:

Relation	Permitted abbreviations
Person	P, P_1, P_2, etc
Company	C, C_1, C_2 etc
jobSkill	J, J_1, J_2 etc
Worksfor	W, W_1, W_2, etc
Knows	K, K_1, K_2, etc
PersonSkill	S, S_1, S_2, etc

Submit your RA expressions for these queries in a .pdf document. (You are strongly encouraged to use Latex.)

Then, for each such RA expression, write a SQL query (possibly using the WITH statement) that mimics this expression as discussed in Section 1.1. Submit these queries in a .sql file as usual.

1. Find the pid and name of each person who (a) works for a company located in 'Bloomington' and (b) knows as person who lives in 'Chicago'. (Assignment 2, problem 1.)

2. Find each job skill that is not the job skill of any person who works for 'Yahoo' or for 'Netflix'. (Assignment 2, problem 7.)

3. Find the cname of each company which employs at least two different persons who have at least one common jobskill. (Assignment 2, problem 3.)

4. Find the pid and name of each person who knows another person who works for 'Google', but who does not know a person who works at 'Amazon' and has the 'Programming' skill. (Assignment 2, problem 2.)

5. Find the pid and name of each person who works for 'IBM' and who has a higher salary that any person with the 'Databases' skill and who also works at 'IBM'. (Assignment 2, problem 4.)

[4] Each of the problems relates back to a corresponding problem in Assignment 2 or in Assignment 3. You may find it useful to look at the SQL solutions for Assignment 2 and Assignment 3 as they may help you in formulating the queries as RA expressions.

6. Find the pid and name of each person who does not know any person who has a salary strictly above 55000. (Assignment 3, problem 12).

7. Find the pid and name of each person who knows all the persons who (a) work at Netflix, (b) make at least 55000, and (c) are born after 1985. (Assignment 3, problem 13).

8. Find the cname of each company who only employs persons who make less than 55000. (Assignment 3, problem 14).

9. Find the pairs of job skills (s_1, s_2) such that each person with job skill s_1 also has job skill s_2. (Assignment 2, problem 8.)

10. Find each triple (p, c, s) where p is the pid of a person, c is the cname of a company, and s is a skill, such that each person who is known by the person with pid p and who works for company with cname c is a person who has the jobskill s. (Assignment 3, problem 16).

11. Find the pairs of company names (c_1, c_2) such that no person who works for the company with cname c_1 has a higher salary than the salaries of all persons who works for the company with cname c_2. (Assignment 2, problem 9.)

2 Theoretical Problems about RA

12. Consider two RA expressions E_1 and E_2 over the same schema. Furthermore, consider an RA expression F with a schema that is not necessarily the same as that of E_1 and E_2.

    Consider the following `if-then-else` query:

    $$\begin{array}{ll} \texttt{if } F = \emptyset & \texttt{then return } E_1 \\ & \texttt{else return } E_2 \end{array}$$

    So this query evaluates to the expression E_1 if $F = \emptyset$ and to the expression E_2 if $F \neq \emptyset$.

    We can formulate this query in SQL as follows[5]:

```
select e1.*
from   E1 e1
where  not exists (select distinct row() from F)
union
select e2.*
from   E2 e1
where  exists (select distinct row() from F);
```

[5] In this SQL query `E1`, `E2`, and `F` denote SQL queries corresponding to the RA expressions E_1, E_2, and F, respectively.

Incidentally, the query

```
select distinct row() from F
```

returns the empty set if $F = \emptyset$ and returns the tuple `()` if $F \neq \emptyset$.[6] In RA, this query can be written as

$$\pi_{()}(F).$$

I.e., the projection of F on an empty list of attributes.

(a) Write an RA expression, in function of E_1, E_2, and F, that expresses this `if-then-else` statement.

(b) Then express this RA expression in SQL with RA operators.

13. Let `A(x)` be a unary relation that can store a set of integers A. Consider the following boolean SQL query:

```
select not exists(select distinct row() from A) as A_isEmpty;
```

This boolean query returns the constant "`true`" if $A = \emptyset$ and returns the constant "`false`" otherwise. Using the insights you gained from Problem 12, solve the following problems:

Write an RA expression that expresses the above boolean SQL query.

Hint: recall that, in general, a constant value "**a**" can be represented in RA by an expression of the form `(A: a)`. (Here, `A` is some arbitrary attribute name.) Furthermore, recall that we can express `(A: a)` in SQL as "`select a as A`". Thus RA expressions for the constants "`true`" and "`false`" can be the expressions `(A: true)` and `(A: false)`, respectively.

14. Let R be a binary relation over schema (A : `integer`, B : `integer`). We say that R is a transitive relation if whenever (a, b) and (b, c) are tuples in R then (a, c) is also a tuple in R.

Consider an arbitrary binary relation R. Write an RA expression that returns the value "`true`" if R is a transitive relation and returns the value "`false`" otherwise.

15. Consider the query "Find the pid of each person born before 1990 and who has all-but-one job skill of the combined set of the job skills of persons who work for Amazon." So we are looking for the pid of each person who is born before 1990 and who lacks precisely one job skill from the set of job skills of persons who works for Amazon.

[6] The tuple `()` is often times referred to as the *empty tuple*, i.e., the tuple without components. It is akin to the empty string ϵ in the theory of formal languages. I.e., the string wihout alphabet characters.

(a) Consult the lecture on set joins and semijoins. Using the techniques described in that lecture, develop a general RA expression for the "`all-but-one`" set semijoin.

Specifically, let $E_1(A, B)$ and $E_2(B)$ be RA expressions, then you need to develop an RA expression for the expression

$$E_1 \ltimes_{\texttt{all-but-one}} E_2$$

which denotes the set of tuples

$$\{a \mid |E_2 - E_1(a)| = 1\}.$$

where $E_1(a)$ denotes the set $\{b \mid (a, b) \in E_1\}$.

Notice that $E_1 \ltimes_{\texttt{all-but-one}} E_2$ is equal to the set

$$\{a \mid \exists b(b \in (E_2 - E_1(a))) \wedge \neg(\exists b_1 \exists b_2(b_1 \neq b_2 \wedge b_1 \in (E_2 - E_1(a)) \wedge b_2 \in (E_2 - E_1(a))))\}$$

which shows that you do not need the `COUNT` function to express $E_1 \ltimes_{\texttt{all-but-one}} E_2$.

(b) Apply this RA expression to the query mentioned above in 15.