Assignment 5: Relational Algebra

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 in the problems in Part 2. 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.

Before you solve the problems in this section, we briefly review how you can express RA expressions in SQL in a way that closely mimics their RA specifications. (For more detail, consult the lectures relating to RA and joins.)

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

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

Then we can write this query in SQL in a variety of ways that closely mimics 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 (B = C)
        WHERE  A = 1) q
```

An alternative way to write this query is to use the `WITH` statement of SQL.[1] To do this, we separate the RA expression F into sub-expressions as follows. (In this case, 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.)

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

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

Then we write the following SQL query. Notice how the expressions $E1$, $E2$, $E3$, and $E4$ occur as separate queries in the WITH statement and that the final query gives the result for the expression F.[2]

```
WITH
E1 AS (SELECT DISTINCT A FROM R),
E2 AS (SELECT DISTINCT A, B, C FROM (R JOIN S ON (B = 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.)

1 Theoretical Problems about RA

1. (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-two`" set semijoin.

   (b) Apply this RA expression to the query "Find the bookno and title of each book that is bought by all but two students who major in 'CS'.

   (c) Formulate the RA expression obtained in Problem 1b in SQL with relational operators. (So no SQL set predicates are allowed in your solution.)

2. 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:

   $$\texttt{if } F \neq \emptyset \quad \texttt{then return } E_1 \quad \texttt{else return } E_2$$

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

[2] 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.

We can formulate this query in SQL as follows[3]:

```
select e1.*
from   E1 e1
where  exists (select 1 from F)
union
select e2.*
from   E2 e1
where  not exists (select 1 from F);
```

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

ii. Then express this RA expression in SQL with RA operators. In particular, you can not use SQL set predicates in your solution.

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

```
select exists(select 1 from A) as A_isNotEmpty;
```

This boolean query returns the constant "`true`" if $A \neq \emptyset$ and returns the constant "`false`" otherwise. Using the insights you gained from Problem 2a, solve the following problems:

i. 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 `(C: a)`. (Here, `C` is some arbitrary attribute name.) Furthermore, recall that we can express `(C: a)` in SQL as "`select a as C`". Thus RA expressions for the constants "`true`" and "`false`" can be the expressions `(C: true)` and `(C: false)`, respectively.

ii. Write a SQL query with relational operators, thus without set predicates, that expresses the above boolean SQL query.

3. Let $f : A \to B$ be a function from a set A to a set B and let $g : B \to C$ be a function from a set B to a set C. The *composition* of the functions f and g, denoted $g \circ f$, is a function from A to C such that for $x \in A$, $g \circ f(x)$ is defined as the value $g(f(x))$.

Represent f in a binary relation `f` with schema `(A,B)` and represent g in a binary relation `g` with schema `(B,C)`.

(a) Write an RA expression that computes the function $g \circ f$. I.e., your expression should compute the binary relation $\{(x, g \circ f(x)) \mid x \in A\}$.

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

(b) Let y be a value in C. Write an RA expression that computes the set $\{x \in A \mid g \circ f(x) = y\}$. I.e., these are the values in A that are mapped by the function $g \circ f$ to the value y.

4. Let $f : A \to B$ be a function from a set A to a set B. We say that f is a *one-to-one* function if for each pair x_1 and x_2 of different values in A (i.e., $x_1 \neq x_2$) it is the case that $f(x_1) \neq f(x_2)$. Represent f by a relation `f` with schema `(A,B)`.

   Write an RA expression that returns the value "`true`" if f (as stored in `f`) is a one-one-one function, and returns the value "`false`" otherwise.

5. Let $f : A \to B$ be a function from a set A to a set B. We say that f is an *onto* function if for each value y in B, there exists a value x in A such that $f(x) = y$. Represent f by a relation `f` with schema `(A,B)`.

   Write an RA expression that returns the value "`true`" if f (as stored in `f`) is an onto function, and returns the value "`false`" otherwise.

6. A *graph* G is a structure (V, E) where V is a set of vertices and wherein E is a set of edges between these vertices. Thus $E \subseteq V \times V$.

   A *path* in G is a sequence of vertices $(v_0, v_1, \ldots, v_n)$ such that for each $i \in [0, n-1]$, $(v_i, v_{i+1}) \in E$. We call n the *length* of this path.

   Represent E by a binary relation `E(source,target)`. A pair (s, t) is in `E` if s and t are vertices in V and (s, t) is an edge in E. Think of s as the source of this edge and t as the target of this edge.

   We say that two vertices v and w in V are connected in G by a path of length n if there exists a path $(v_0, v_1, \ldots, v_n)$ such that $v = v_0$ and $w = v_n$.

   Write an RA expression that returns the set of pairs (v, w) that are connected by a path of length at most n. (You may assume that $n \geq 1$.)

2 Formulating Queries in RA

In the following problems, we will use the data that you can find in the `data.sql` file provided for these problems.

Write the following queries as RA expressions in the standard RA notation. Submit these queries in a .pdf document. In these expressions, you can use the following notations for the relations:

Student	S, S_1, S_2, etc
Book	B, B_1, B_2 etc
Cites	C, C_1, C_2 etc
Major	M, M_1, M_2, etc
Buys	T, T_1, T_2, etc

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

Furthermore, and where possible, avoid using $\times$ or `CROSS JOIN` operator. In addition, where possible, using semijoin operations instead of join operations.

Each of the problem relates back to a problem in Assignment 2. You can consult the SQL solutions for these problem as they may help you in formulating the queries as RA expressions.

7. Find the sid and name of each student who majors in CS and who bought a book that cost more than $10. (Assignment 2, Problem 1.)

8. Find the bookno, title, and price of each book that cites at least two books that cost less than $60. (Assignment 2, Problem 3.)

9. Find the bookno, title, and price of each book that was not bought by any Math student. (Assignment 2, Problem 2.)

10. Find the sid and name of each student along with the title and price of the most expensive book(s) bought by that student. (Assignment 2, Problem 4.)

11. Find the booknos and titles of books with the next to highest price. (Assignment 2, Problem 6.)

12. Find the bookno, title, and price of each book that cites a book which is not among the most expensive books. (Assignment 2, Problem 7.)

13. Find the sid and name of each student who has a single major and such that none of the book(s) bought by that student cost less than $40. (Assignment 2, Problem 8.)

14. Find the bookno and title of each book that is bought by all students who major in both “CS” and in “Math”. (Assignment 2, Problem 9.)

15. Find the sid and name of each student who, if he or she bought a book that cost at least than $70, also bought a book that cost less than $30. (Assignment 2, Problem 10.)

16. Find each pair (s1, s2) where s1 and s2 are the sids of students who have a common major but who did not buy the same set of books. (Assignment 2, Problem 11.)