Assignment 3

Due Thursday February 7, 2019 by 11:45pm EST

In this assignment, you will be required to use PostgreSQL. Your solutions should include the PostgreSQL statements for solving the problems. Turn in a `.sql` file with your solutions on IUCANVAS.[1] Where useful, we encourage you to include comments to explain your solutions.

1 Queries with expressions and functions; Boolean queries

For the problems in this section, you can use views (including parameterized views, i.e., user-defined functions) but you can not use aggregate functions nor the `GROUP BY` and `HAVING` clauses. You can also not use the `INNER JOIN` (or other joins) operators.

1. Let $A(x)$ be a unary relation schema that represent a set of non-negative integers. (I.e., attribute x has domain `INTEGER`.) Write a SQL statement that produces a table that, for each $x \in A$, list the tuple $(x, \sqrt{x}, x^2, 2^x, x!, \ln x)$.[2]

   For example, if $A = \{1, 2, 3, 4, 5\}$ then your SQL statement should produce the following table:

```
  x |  square_root_x   | x_squared | two_to_the_power_x | x_factorial |    logarithm_x
---+------------------+-----------+--------------------+-------------+-------------------
  1 |                1 |         1 |                  2 |           1 |                 0
  2 |  1.4142135623731 |         4 |                  4 |           2 | 0.693147180559945<
  3 | 1.73205080756888 |         9 |                  8 |           6 |  1.09861228866811
  4 |                2 |        16 |                 16 |          24 |  1.38629436111989
  5 | 2.23606797749979 |        25 |                 32 |         120 |   1.6094379124341
( 5 rows)
```

2. Let $A(x)$, $B(x)$ and C be three unary relation schemas that represent sets A, B and C of integers. (I.e., the domain of the attribute x is `INTEGER`.)

   For each of the following subproblems, write two SQL statements to determine the specified property on the sets A, B, and C. (Your solutions should work for arbitrary sets A, B, and C.)

   In the first SQL statement, you should make appropriate use of the set operations `UNION`, `INTERSECT`, or `EXCEPT`.

   In the second SQL statement, you are not allowed to use these set operators. Instead, you need to use the set predicates `IN`, `NOT IN`, `EXISTS`, or `NOT EXISTS`.

   (a) Determine whether or not $A \cap B = \emptyset$.

   For example, if $A = \{1, 2\}$ and $B = \{1, 4, 5\}$ then the result of your statement should be

[1] Use the standard turn-in procedure for assignments.

[2] For this problem, you can use pre-defined functions supported by PostgreSQL.

```
 answer
--------
 f
(1 row)
```

If, however, $A = \{1, 2\}$ and $B = \{3, 4\}$ then the result of your statement should be

```
 answer
--------
 t
(1 row)
```

(b) Determine whether or not $A \neq B$.

(c) Determine whether or not $(A \cap B) \supseteq C$.

3. Consider the relation schema Point(pid, x, y) of a relation of points in the plane. The attribute pid (of type INTEGER) is the identifier of a point, and the attributes x and y, both of type FLOAT, are its x and y coordinates.

   Write a SQL query that returns the (p_1, p_2) pairs of different pids of points that are closest in distance from each other. Recall that if $p_1 = (x_1, y_1)$ and $p_2(x_2, y_2)$ are two points in the plane, then the distance between them is given by the formula

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

   For example if you have the following points,

```
pid | x | y
----+---+---
  1 | 0 | 0
  2 | 0 | 1
  3 | 1 | 0
```

   Then your answer should be

```
p1  | p2
----+---+---
1   | 2
2   | 1
1   | 3
3   | 1
```

2 Queries using aggregate functions

In the problems in this section, you will practice working with aggregate functions.

4. Let $p(x)$ and $q(x)$ be 2 polynomials with integer coefficients.

   Let `P(coefficient, degree)` and `Q(coefficient, degree)` be two binary relations representing $p(x)$ and $q(x)$, respectively. E.g., if $p(x) = 3x^3 - 2x^2 + 5$ then its representation in the relation `P` is as follows:

   P

coefficient	degree
3	3
-2	2
5	0

   Write a SQL statement that computes a binary relation representing the multiplication of $p(x)$ and $q(x)$, i.e., the polynomial $p(x) * q(x)$.

   For example, consider $p(x) = 2x^2 - 5x + 5$ and $q(x) = 3x^3 + x^2 - x$. Then $p(x) * q(x) = 6x^5 + (2 - 15)x^4 + (-2 - 5 + 15)x^3 + (5 + 5)x^2 - 5x = 6x^5 - 13x^4 + 8x^3 + 10x^2 - 5x$. So, for these polynomials, your SQL query should return the relation

coefficient	degree
6	5
-13	4
8	3
10	2
-5	1

   Your solution should work for arbitrary polynomials $p(x)$ and $q(x)$.

   (`Hint`: For this problem, you must use the SUM aggregate function.)

5. Let M be an $n \times n$ matrix of integers (n is a positive integer).

   For $i, j \in [1, n]$, we will denote by $M[i, j]$ the element in matrix M at row i and column j.

   Given two $n \times n$ matrix M and N, denotes by $M \cdot N$ the matrix multiplication of M and N. By definition, $M \cdot N$ is again a $n \times n$ matrix where, for $i, j \in [1, n]$, row i and column j of $M \cdot N$ is defined by

$$M \cdot N[i, j] = \sum_{k=1}^{n} M[i, k]N[k, j].$$

   The matrix M can be represented using a relation `M` with schema (`row`, `colmn`, `value`), and similarly for the matrix N.[3]

   For example if M is the 3×3 matrix

$$M = \begin{array}{ccc} 1 & 2 & 3 \\ 1 & -3 & 5 \\ 4 & 0 & -2 \end{array}$$

   then `M` is the following relation of 9 tuples:

M		
`row`	`colmn`	`value`
1	1	1
1	2	2
1	3	3
2	1	1
2	2	-3
2	3	5
3	1	4
3	2	0
3	3	-2

   Let M and N be two $n \times n$ matrices represented by the two relations `M` and `N`.

   Write a SQL query that computes a relation over schema (`row`, `column`, `value`) that represents the matrix $M \cdot N$.

   (**Hint:** use the SQL aggregate function `SUM`.)

   Your solution should work for any $n \geq 1$.

[3]Notice that we use the attribute name '`colmn`' since the word '`column`' is a reserved word in PostgreSQL.

For the following problems, use the relations `student`, `major`, `book`, and and `buys` that can be found in the `data.sql` file.

6. Write a function `booksBoughtbyStudent(sid int)` that returns the table of all the books bought by that student.

   Using this function, write the following queries:

   (a) Find for each sid of a student, the number of books that cost more than \$40 and that were bought by that student.

   (b) Find the sids and names of the students who spent the most on the books that they bought.

   (c) Find for each sid of a student the booknos and titles of the least expensive books bought by that student.

   (d) Find the pairs of sids (s_1, s_2) of different students who bought the same number of books.

3 Queries with quantifiers using Venn diagrams with condition

For the following problems, use the relations `student`, `major`, `book`, and `buys` that can be found in the `data.sql` file.

Using the method of Venn diagrams with conditions and without using the `COUNT` function, write SQL queries for the following queries with quantifiers.

In these problems, you must write appropriate views and parameterized views for the sets A and B that occur in the Venn diagram with conditions for these queries. (See the lecture on Queries with Quantifiers.)

7. Find the sid and name of each student who bought no books that cost more than \$60.

8. Find the bookno and title of each book bought by all the students who major in both 'CS' and in 'Math'.

9. Find the sid and name of each student who bought some of the least expensive books.

10. Find the pairs of sids (s_1, s_2) of different students such that student s_1 bought not only books bought by the student with sid s_2.

4 Queries with quantifiers using Venn diagrams with counting conditions

Using the method of Venn diagram with counting conditions, write SQL queries for the following queries with quantifiers.

In these problems, you should write appropriate views and parameterized views for the sets A and B that occur in the Venn diagrams for these queries. (See the lecture on Queries with Quantifiers Using the COUNT function.)

11. Find the bookno and title of each book that was bought by more than two students who major in 'CS'.
12. Find the sid and name of each student who bought an odd number of books that cost more than \$35.
13. Find the bookno and title of each book that was bought by all but 3 students.
14. Find the pairs (s_1, s_2) of different students who did not buy the same books.

5 Simulating Set Predicates with the COUNT Function

In the following problems you will need to write SQL queries wherein the set predicates are simulated using the SQL `COUNT` function. (See the lecture Simulating Set Predicates Using COUNT.)

Rewrite each of the following SQL queries into an equivalent SQL query wherein the set predicates are simulated using the `COUNT` function.

15.
```
SELECT s.sid, s.sname
FROM   Student s
WHERE  EXISTS(SELECT b.bookno
              FROM   Book b
              WHERE  b.title  <> 'Networks' AND
                     b.title = SOME (SELECT b1.title
                                     FROM   Buys t, Book b1
                                     WHERE  t.sid = s.sid AND t.bookno = b1.bookno));
```

16.
```
SELECT b.bookno, b.title
FROM   Book b
WHERE  b.price <= ALL (SELECT b1.price
                       FROM   Book b1
                       WHERE  b1.bookno NOT IN (SELECT t.bookno
                                                FROM   Buys t, Major m
                                                WHERE   t.sid = m.sid and m.major = 'CS'));
```