CSE 544 Homework 3
Due date: Thursday, March 10, 2011

1 Transactions

1. Refer to the textbook pp. 574, exercise 17.2. For each scheulde in questions 6, 8, 10, 11, 12, determine whether the schedule is (a) conflict serializable, (b) view-serializable, (c) recoverable, and (d) avoids cascading aborts.

2. Textbook pp. 598, exercises 18.4 and 18.5.

2 Query Plans

In this problem we will enumerate join plans. For counting purposes we assume the join operator is not commutative, i.e. the plans $R \bowtie S$ and $S \bowtie R$ are consider to be distinct.

1. Consider the query $R_1(A_0, A_1) \bowtie R_2(A_1, A_2) \bowtie \ldots \bowtie R_n(A_{n-1}, A_n)$ (all joins are natural joins). How many distinct left-deep join plans without cartesian products are there for this query ?

2. Consider the query $S(A_1, A_2, \ldots, A_n) \bowtie R_1(A_1, B_1) \bowtie \ldots \bowtie R_n(A_n, B_n)$. How many distinct bushy join plans without cartesian products are there for this query ?

3 Relational Calculus

1. This is the famous drinkers-beers-bars problem, used by Ullman in his early textbook on databases. Consider the following schema:

`Likes(drinker, beer), Frequents(drinker, bar), Serves(bar, beer)`

We will abbreviate the table names with L, F, S. For example the following query finds all drinkers that like only `Bud-Light`:

$$q(d) \quad :- \quad (\exists b.L(d,b)) \wedge (\forall b.L(d,b) \Rightarrow b = \texttt{Bud-Light})$$

Note that the first condition ensures that the query is domain independent.

For each of the questions below, write a relational query and then an equivalent relational algebra plan; draw the plan as a tree.

(a) Find all drinkers that frequent only bars that serve only beer they like. (Optimists)

(b) Find all drinkers that frequent only bars that serve some beer they like. (Realists)

(c) Find all drinkers that frequent some bar that serves only beers they like. (Prudents)

(d) Find all drinkers that frequent only bars that serve none of the beers they like. (Flagellators)

2. Consider the vocabulary $L(x,y), A(x), B(x)$ that represents two bit strings of length n as follows:

   - $L(x,y)$ is a strict, total order over an active domain of size n, meaning that it satisfies the following constraints: anti-reflexive $\neg L(x,x)$, anti-symmetric $\neg(L(x,y) \wedge L(y,x))$ and transitife $L(x,y) \wedge L(y,z) \Rightarrow L(x,z)$.
   - $A(x), B(y)$ denote two input strings in $\{0,1\}^n$.

   For example, if $n = 3$, $A = 101$ and $B = 110$, then define $L = \{(u,v),(u,w),(v,w)\}$, $A = \{u,w\}$, $B = \{v,w\}$.

   (a) Write a relational query to compute $C = A + B$. Your query $q(x)$ returns all position x in C that are 1. You will ignore the overflow bit, since you don't have a symbol for its position: for example, if $A = 101$ and $B = 110$ then you should return $C = 011$ (in other words, $q(x)$ returns $\{u,v\}$).

(b) **This question requires some research** Is it possible to write multiplication in the relational calculus ? The query $q(x, y)$ needs two head variables, because the product of two strings of length n is a string of length n^2. You need to research your answer and argue whether multiplication is possible, or not possible in the relational calculus.

4 Datalog

1. Consider a ternary relation $T(x, y, z)$ representing a graph, where each node x has either zero or two outgoing edges $(x, y), (x, z)$. Note that x is a key in $T(x, y, q)$. Consider the following game with two players. Players take turns in moving a pebble on the graph. If the pebble is on a node x, then the player whose turn it is may move it to one of the two children, y or z. The player who cannot move (because the current node has no children), loses. Write a datalog program to compute for each node whether player 1 or player 2 has a winning strategy. Assume $L(x)$ is a predicate containing all leaf nodes, i.e. $L(x) \equiv \neg\exists y.\exists z.T(x, y, z)$.

2. **Challenging question** Now consider the same game on a graph given by a binary relation $E(x, y)$; that is, a node x may have an arbitrary number of children; as before $L(x)$ denotes the set of leaves. Write a datalog$^\neg$ program to compute for each node whether player 1 or player 2 has a winning strategy.

3. We discussed two possible semantics for datalog$^\neg$ in class. In *stratified semantics* the rules of the datalog program are partitioned into strata such that the IDBs occurring in strata k may only occur negated in bodies of rules belonging to strata $> k$. In *inflationary semantics*, we compute the datalog program such that at iteration $i + 1$ we extend the content of the IDBs from iteration i with new facts (i.e. we never remove facts from IDBs). Suppose a datalog$^\neg$ program can be stratified. Do the two semantics coincide ? If not, then give a counterexample of a datalog$^\neg$ program and EDBs such that the two semantics return two different answers.

4. (Exercise 13.14 in AHV) Perform a magic-set optimization for the following datalog query:

```
sgv(x,y) :- flat(x,y)
sgv(x,y) :- up(x,z1), sgv(z1,z2), flat(z2,z3), sgv(z3, z4),down(z4,y)
query(y) :- sgv(a,y)
```

5. **This question requires some research** Fix a vocabulary $E(x, y)$ where each instance represents a graph. We have shown in class that every datalog$^\neg$ query can be computed in PTIME in the size of the graph. Is it the case that any graph property that can be computed in polynomial time, can also be expressed in datalog$^\neg$? If your answer is *false* then give a counterexample. You do not need to provide proofs, i.e. you can either answer *true*, or you can answer *false* and give a counterexample without proving that it is a counterexample.

5 Conjunctive Queries

1. For each of the queries below, find a semijoin reducer. If the query admits a full reducer then your answer should be a full reducer; otherwise your answer can be any reducer that is not full, and you need to indicate that the query does not admit a full reducer.

$$q_1(x) : -R(x, y), S(y, z), T(y, u)$$

$$q_2(x) : -R(x, y), S(y, z), T(z, u), M(z, v)$$

$$q_3(x) : -R(x, y), S(y, z), T(z, u), M(u, y)$$

2. Indicate for each pair of queries q, q' below, whether $q \subseteq q'$. If the answer is yes, provide a proof; if the answer is no, give a database instance I on which $q(I) \not\subseteq q'(I)$.

(a)

$$\begin{aligned} q(x) &: - \quad R(x, y), R(y, z), R(z, x) \\ q'(x) &: - \quad R(x, y), R(y, z), R(z, u), R(u, v), R(v, z) \end{aligned}$$

(b)

$$\begin{aligned} q(x, y) &: - \quad R(x, u, u), R(u, v, w), R(w, w, y) \\ q'(x, y) &: - \quad R(x, u, v), R(v, v, v), R(v, w, y) \end{aligned}$$

(c)

$$\begin{aligned} q() &:- R(u,u,x,y), R(x,y,v,w), v \neq w \\ q'() &:- R(u,u,x,y), x \neq y \end{aligned}$$

(d)

$$\begin{aligned} q(x) &:- R(x,y), R(y,z), R(z,v) \\ q'(x) &:- R(x,y), R(y,z), y \neq z \end{aligned}$$

3. Consider the two conjunctive queries below, and notice that $q_1 \subset q_2$.

$$\begin{aligned} q_1(x) =& R(x,y), R(y,z) \\ q_2(x) =& R(x,y) \end{aligned}$$

   (a) Find a conjunctive query $r(x)$ s.t. $q_1 \subset r \subset q_2$

   (b) **Challenging question** Extend your answer: find an infinite set of queries $r_1(x), r_2(x), \ldots$ such they are inequivalent $r_i \not\equiv r_j$ for $i \neq j$, and for every i, $q_i \subset r_i \subset q_2$. If you answer this question then you do not need to answer the preceding question.

6 Provenance

Consider a relational database schema $R(A), S(B,C)$. All queries mentioned below are assumed to be monotone queries.

1. One of the answers a of a relational query Q has the provenance polynomial $x_1y_1^2 + 2x_2y_1y_2$, where x_1, x_2 are annotations of two tuples in R and y_1, y_2 are annotations of two tuples in S.

   (a) Assume that the input relations are sets. If we evaluate the query under bag semantics, how many copies of a will be in the query's answer ?

   (b) Assume that the input relations are bags, where each tuple occurs exactly twice. If we evaluate Q under bag semantics, how many copies of a are there in the query's answer ?

(c) What is the smallest number of tuples that need to be removed from the database instance in order to remove a from the answer to Q ?

(d) Suppose that the tuple annotated with y_2 was incorrect (it contains some incorrect value). Is the answer a correct, or did it become incorrect ?

2. Consider the following instance:

R

A	
a_1	x_1
a_2	x_2

S

A	B	
a_1	b_1	y_1
a_1	b_2	y_2
a_2	b_2	y_3

For each polynomial below, write a Boolean conjunctive query having that provenance polynomial. Your queries should not have constants or the inequality predicates $\neq, <, \leq$.

$$
\begin{aligned}
P_1 =& x_1y_1 + x_1y_2 + x_2y_3 \\
P_2 =& x_1y_1^2 + x_1y_2^2 + x_1y_2y_3 + x_2y_2y_3 + x_2y_3^2 \\
P_3 =& x_1^2y_1^2 + x_1^2y_2^2 + 2x_1x_2y_2y_3 + x_2^2y_3^2
\end{aligned}
$$

3. Now we consider arbitrary instances R, S, and we assume that the tuples in R are annotated with variables $x_1, x_2, \ldots$ and the tuples in S are annotated with $y_1, y_2, \ldots$ In each case below give an example of a Boolean conjunctive query whose provenance polynomial P has the property stated below. Your conjunctive query may use the predicate $\neq$.

(a) The polynomial P factorizes as $P = P_1 \cdot P_2$ where P_1 is a linear polynomial in $x_1, x_2, \ldots$ and P_2 is a linear polynomial in $y_1, y_2, \ldots$

(b) All monomials in P have the form $x_iy_jy_k$ where $j \neq k$.

(c) All monomials in P have the form $x_ix_jy_k$ where $i \neq j$.