

---

On the cover page of your assignment, you must list everyone with whom you discussed this assignment, and which problems you discussed with each person. You must also write **and sign** the following statement: “I have read and understood the policy on collaboration on homework assignments stated in the Course Information handout.” Without these, your homework will not be marked.

---

1. Reformulate each of the following queries so that they are safe: [10]

- (a)  $\langle (x, y) : x \neq y \wedge R(x, y) \rangle$ ;
- (b)  $\langle x : \neg(P(x) \vee \neg Q(x)) \rangle$ ;
- (c)  $\langle x : \forall y \neg R(x, y) \supset P(x) \rangle$ ;
- (d)  $\langle (x, y) : \neg[S(x, y) \supset \forall z(R(x, z) \wedge Q(y, z) \supset z = a)] \rangle$ ;
- (e)  $\langle x : P(x) \wedge (\exists y S(x, y) \equiv \exists z T(z, x)) \rangle$ ;

2. For each of the following queries, prove that it is safe or prove that it is not safe, and state the answer to the query for the relational database about instructors, students, grades *etc.* presented in class. [25]

- (a)  $\langle (x, y) : \exists u(Enrolled(x, u) \wedge Prereq(y, u)) \vee \neg \exists z(Grade(x, y, z) \wedge z \neq a) \rangle$
- (b)  $\langle (x, y) : \exists u(Enrolled(x, u) \wedge Prereq(y, u)) \wedge \neg \exists z(Grade(x, y, z) \vee z = a) \rangle$
- (c)  $\langle (x, y) : \exists u(Enrolled(x, u) \wedge Prereq(cs148, u)) \wedge \neg \exists z(Grade(x, y, z) \wedge z \neq a) \rangle$
- (d)  $\langle (x, y) : \exists u(Enrolled(x, u) \wedge Prereq(y, u)) \vee \exists z(Grade(x, y, z) \wedge z \neq a) \rangle$
- (e)  $\langle (x, y) : \exists u(Enrolled(x, u) \wedge Prereq(cs148, u)) \vee \exists z(Grade(x, y, z) \wedge z \neq a) \rangle$

3. In this question, you will use Prolog to answer queries about a relational database. First make up a simple database about songs and musicians (not necessarily real ones) using the following predicates: [35]

**musician**( $p, r, s$ )   musician  $p$  played role  $r$  on song  $s$   
(a role might be “vocals” or “lead guitar”);  
**song**( $s, y, l$ )   song  $s$  was recorded in year  $y$  and has length  $l$   
(for some suitable units of duration).

Now formulate each of the following questions as a safe query using Prolog’s notation, pose it to Prolog, and obtain Prolog’s answer:

- (a) Who played organ on “Hey Jude”?
- (b) What songs recorded in 2004 were solos (one musician only)?

- (c) What musicians have played more than two roles on the same song?
- (d) Was “Like a Rolling Stone” the longest song recorded that year? (Assume that you can use special built-in relations like  $<$  and  $>$  on numbers.)
- (e) What songs recorded in the nineties have musicians who played on no other songs?

In the final two parts, define the new relations using Prolog and test them on some examples of your choice.

- (e) `collaborators`( $p_1, p_2$ ),  $p_1$  and  $p_2$  have worked together on some song;
- (f) `duo`( $p_1, p_2$ ), there are two or more songs recorded in the same year where  $p_1$  and  $p_2$  are the only musicians.

You should hand in a listing of your Prolog database, and a listing that shows the queries you submitted to Prolog, and the answers returned.

4. Consider the unsafe query  $\langle (x, y) : P(x) \wedge x = y \rangle$ . For each of the queries below, explain whether or not it is safe, and whether or not it would return the same answer as the original query above: [12]

- (a)  $\langle (x, y) : P(x) \wedge P(y) \rangle$ ;
- (b)  $\langle (x, y) : P(x) \wedge P(y) \wedge x = y \rangle$ ;
- (c)  $\langle (x, y) : (P(x) \vee P(y)) \wedge x = y \rangle$ .

5. Let us define two queries of a fixed relational language to be *query equivalent* if they return the same answer for every relational database over this language. Some queries that are not safe can be reformulated in a safe, query equivalent way; some cannot because their answers depend on the global domain of the database (beyond the extensions of the predicates and the denotations of the constants in the query). [18]

Consider the following two queries:

- (a)  $\langle x : P(x) \wedge \forall y \neg (R(x, y) \vee P(y)) \rangle$ ;
- (b)  $\langle x : P(x) \wedge \neg \forall y (R(x, y) \vee P(y)) \rangle$ .

Neither query is safe, because of the  $\forall$  symbol. For each query, either prove that it can be safely reformulated (by producing a new query and proving that it is both safe and query equivalent to the original), or prove that it cannot be safely reformulated (by producing two relational databases with the same interpretation for both predicate symbols, but where the query returns different answers).

**Bonus question:** We investigate the relationship between query equivalence, defined above, and logical equivalence (and validity), as defined in class.

1. Show that logical equivalence implies query equivalence. To be concrete, assume that we have two formulas  $F$  and  $G$  in the same language and with a single free variable  $x$ . Prove that if  $\forall x(F \equiv G)$  is true in every structure (valid), then  $\langle x : F \rangle$  and  $\langle x : G \rangle$  are query equivalent.
2. Prove that the converse to part (a) does not hold. *Hint:* In Section 4 of the course (Slide 18), there is a sentence that is false in every structure with a finite domain, but is true in some structures with an infinite domain. Use it in a query.

*Do not attempt any bonus work until the regular part of your assignment is complete.*