Name

CWID

Midterm Exam

March 12th, 2020
10:00-11:15

CS520 - Data Integration, Warehousing, and Provenance

Results

Please leave this empty!

1.1 1.2 1.3

Sum

Instructions

- Try to answer all the questions using what you have learned in class. Keep hard questions until the end.
- **When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!**
- The exam is closed book and closed notes! No calculator, smartphones, or similar allowed!

Consider the following database schema and example instance about music albums:

user

nickname	name	postVisibility	country
BobAwesome	Bob	FOF	USA
Ali12	Alice	friends	France
Peter	Peter	friends	India
Pokegert	Gert	public	China

friends

person	friend
BobAwesome	Ali12
Ali12	BobAwesome
BobAwesome	Peter
Peter	Pokegert

posts

pid	user	text	time
1	BobAwesome	Hello just brought ...	2018-01-10
2	BobAwesome	meet @Ali12 at ...	2018-01-11
3	Peter	... is great, would recommend.	2018-01-15

Hints:

- Attributes with black background form the primary key of a relation (e.g., *nickname* for relation *user*)
- The attributes *person* and *friend* of relation *friends* are a foreign key to relation *user*.
- The attribute *user* of relation *posts* is a foreign key to relation *user*.

Part 1.1 Datalog (Total: 38 Points)

Recall that Datalog applies set semantics.

Question 1.1.1 (5 Points)

Write a **Datalog program** that returns the *name* and *nickname* of users from USA.

Solution

```
Q(NN,N) :- user(N,N,_,usa).
```

Question 1.1.2 (7 Points)

Write a **Datalog program** that returns the *names* of users which are living in France and have posted before 2018-01-10 or live in USA and whose posts are visible to the public (post visibility = public).

Solution

```
Q(X) :- user(N,X,_,france), posts(_,N,_,T), T < 2018-01-10.
Q(X) :- user(_,X,public,usa).
```

Question 1.1.3 (8 Points)

Write a **Datalog program** that returns the names of users that are not friends of user *BobAwesome* nor are they friends of a friend of *BobAwesome*. For example, in the example EDB instance there is no such person.

Solution

```
friendsOfBob(X) :- friends(bobawesome,X).
friendsOfBob(X) :- friends(bobawesome,Y), friends(Y,X).
Q(X) :- users(X,_,_,_), not friendsOfBob(X).
```

Question 1.1.4 (9 Points)

Write a **Datalog program** that returns users who have friends or friends of friends in every country.

Solution

```
fofCntr(X,C) :- friends(X,Y), user(Y,_,_,C).
fofCntr(X,C) :- friends(X,Z), friends(Z,Y), user(Y,_,_,C).
countries(X) :- user(_,_,_,X).
U(X) :- user(X,_,_,_).
missingCountry(X) :- , country(C), not fofCntr(X,C).
Q(X) :- U(X), not missingCountry(X).
```

Question 1.1.5 (9 Points)

Write a **Datalog program** that returns pairs of countries (C1,C2) such that there exists at least one path in the friendship graph that connects a user from country C1 with a user from country C2. Here we do not care about the direction of edges, e.g., there is a path from Pokegert to Ali12.

Solution

```
TC(X,Y) :- friends(X,Y).
TC(X,Y) :- friends(Y,X).
TC(X,Y) :- TC(X,Z), TC(Z,Y).
Q(C1,C2) :- user(X,_,_,C1), user(Y,_,_,C2), TC(X,Y).
```

Part 1.2 Constraints (Total: 26 Points)

Question 1.2.1 Expressing Constraints in First-Order Logic (13 Points)

Recall the representation of constraints as universally quantified formulas in first-order logic introduced in class. Write down the logical encoding of the following constraints over the example schema:

- The foreign key from attribute friend of relation friends to relation user.
- Friendship has to be reciprocal, i.e., if X is a friend of Y, then also Y has to be a friend of X.
- The primary key of relation posts
- The following functional dependency for relation users: $country \rightarrow postVisibiility$

Solution

$$
\begin{aligned}
FK_1 :& \forall p, f : friends(p, f) \rightarrow \exists x_1, x_2, x_3 : user(f, x_1, x_2, x_3) \\
REPROC :& \forall x, y friends(x, y) \rightarrow friends(y, x) \\
PK :& \forall p, u_1, t_1, i_1, p, f_2, t_2, i_2 : posts(p, u_1, t_1, i_1) \wedge posts(p, u_2, t_2, i_2) \rightarrow u_1 = u_2 \wedge t_1 = t_2 \wedge i_1 = i_2 \\
FD :& \forall n_1, a_1, p_1, c, n_2, a_2, p_2 : user(n_1, a_1, p_1, c), user(n_2, a_2, p_2, c) \rightarrow p_1 = p_2
\end{aligned}
$$

Question 1.2.2 Creating Denial Constraints (13 Points)

Create denial constraints over the example schema based on the following descriptions.

- The friendship graph is not allowed to contain any triangles, i.e., this constraint is violated if there exists users X, Y, and Z such that X is a friend of Y, Y is a friend of Z, and Z is a friend of X (Note that the direction of edges matters!.
- Users of country USA are not allowed to post after 2020-03-11.
- Implement the primary key of relation users.

Solution

$$d_1 : \forall u_1, u_2, u_3 : \neg(friends(u_1, u_2) \wedge friends(u_2, u_3) \wedge friends(u_3, u_1))$$

$$d_2 : \forall n, a, p, e, t : \neg(user(n, a, p, \mathsf{USA}) \wedge posts(p, n, e, t) \wedge t > 2020 - 03 - 11)$$

$$\begin{aligned} d_3 : &\forall u, n_1, p_1, c_1, n_2, p_2, c_2 : \neg(user(u, n_1, p_1, c_1) \wedge user(u, n_2, p_2, c_2) \wedge n_1 \neq n_2) \\ &\forall u, n_1, p_1, c_1, n_2, p_2, c_2 : \neg(user(u, n_1, p_1, c_1) \wedge user(u, n_2, p_2, c_2) \wedge p_1 \neq p_2) \\ &\forall u, n_1, p_1, c_1, n_2, p_2, c_2 : \neg(user(u, n_1, p_1, c_1) \wedge user(u, n_2, p_2, c_2) \wedge c_1 \neq c_2) \end{aligned}$$

Part 1.3 Query Containment And Equivalence (Total: 36 Points)

Question 1.3.1 (36 Points)

Consider the queries shown below. Check all possible containment relationships. If there exists a containment mapping from Q_i to Q_j then write down the mapping.

```
Q1(X,Y) :- R(X,Z), S(Z,A), R(A,Y).
Q2(Y,X) :- R(Z,X), R(Y,B), R(C,B), R(D,B).
Q3(X,Y) :- R(X,Z), R(A,Y).
Q4(A,B) :- T(X,Z), R(A,X), R(Y,B), U(Z,Y).
```

Solution

$\underline{Q_1 \to Q_2}$:
no containment mapping exists

$\underline{Q_1 \to Q_3}$:
no containment mapping exists

$\underline{Q_1 \to Q_4}$:
no containment mapping exists

$\underline{Q_2 \to Q_1}$:

$$X \to Y$$
$$Y \to X$$
$$Z \to A$$
$$B \to Z$$
$$C \to X$$
$$D \to X$$

$\underline{Q_2 \to Q_3}$:

$$X \to Y$$
$$Y \to X$$
$$Z \to A$$
$$B \to Z$$
$$C \to X$$
$$D \to X$$

$\underline{Q_2 \to Q_4}$:

$$X \to B$$
$$Y \to A$$
$$Z \to Y$$
$$B \to X$$
$$C \to A$$
$$D \to A$$

$\underline{Q_3 \to Q_1}$:

$$X \to X$$
$$Y \to Y$$
$$Z \to Z$$
$$A \to A$$

$\underline{Q_3 \to Q_2}$:

$$X \to Y$$
$$Y \to X$$
$$Z \to B$$
$$A \to Z$$

$\underline{Q_3 \to Q_4}$:

$$X \to A$$
$$Y \to B$$
$$Z \to X$$
$$A \to Y$$

$\underline{Q_4 \to Q_1}$:
no containment mapping exists

$\underline{Q_4 \to Q_2}$:
no containment mapping exists

$\underline{Q_4 \to Q_3}$:
no containment mapping exists