Name

CWID

Midterm Exam

October 12, 2023
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 buyers, credit cards, orders, and products.

buyer

name	age	gender
alice	20	female
bob	21	male
carol	18	female

card

cardNum	owner	limit
1111	alice	50
1234	bob	10
4321	bob	30
9999	carol	1000

product

pname	type	price	weight
pen	office	3	5
pencil	office	2	3
notebook	office	10	400
camera	electronic	300	600
bike	transport	100	15000
skateboard	transport	50	1500
pan	kichen	25	700

order

buyer	product	count
alice	pen	4
alice	notebook	2
bob	bike	1
alice	pan	1
carol	camera	1
carol	skateboard	1

Hints:

- Attributes with black background form the primary key of a relation
- The attribute *buyer* of relation *order* is a foreign key to relation *buyer*. The attribute *product* of relation *order* is a foreign key to relation *product*.
- The attribute *owner* of relation *card* is a foreign key to relation *buyer*.

Part 1.1 Datalog (Total: 38 Points)

Recall that Datalog applies set semantics. All Datalog questions use the schema shown above.

Question 1.1.1 (5 Points)

Write a **Datalog program** that returns the *pname* and *price* of products of type *office* which weight more than 100 (lb).

Solution

```
q(N,P) :- product(N,office,P,W), W > 100.
```

Question 1.1.2 (6 Points)

Write a **Datalog program** that returns pairs of buyer names and the names of products the buyer has brought. Only include products that cost more than $100 or weight more than 200 (lb).

Solution

```
heavy_or_expensive(N) :- product(N,_,P,_), P > 100.
heavy_or_expensive(N) :- product(N,_,_,W), W > 200.
q(N,P) :-  buyer(N,_,_), order(N,P,_), heavy_or_expensive(P).
```

Question 1.1.3 (8 Points)

Write a **Datalog program** that returns the names of buyers that have only brought products of type *transport*.

Solution

```
has_non_transport(B) :- buyer(B,_,_), order(B,P,_), product(P,T,_,_), T <> transport.
q(B) :- buyer(B,_,_), not has_non_transport(B).
```

Question 1.1.4 (10 Points)

Write a **Datalog program** that returns *pname* and *price* for products that have been bought by every buyer younger than 20 years.

Solution

```
young_buyer(B) :- buyer(B,A,_), A < 20.
product_names(P,R) :- product(P,_,R,_).
possible_order(B,P) :- young_buyer(B), product_names(P).
actual_order(B,P) :- order(B,P,_).
missing_order(P) :- possible_order(B,P), not actual_order(B,P).
q(P,R) :- product_names(P,R), not missing_order(P).
```

Question 1.1.5 (9 Points)

Consider the following graph $G = (V, E)$ where V is the set of buyers and there is an edge (b_1, b_2) from buyer b_1 to buyer b_2 if the orders of b_1 and b_2 have at least one product in common. Write a **Datalog program** that returns pairs of buyers that are connected in this graph by some path. Note that the graph is not given as input, but needs to be computed by your query based on the relations in the database.

Solution

```
e(X,Y) :- buyer(X,_,_), buyers(Y,_,_), order(X,P,_), order(Y,P,_).
path(X,Y) :- e(X,Y).
path(X,Y) :- path(X,Z), e(Z,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 first-order logic implications as introduced in class. Write down the following constraints over the example schema as universally quantified formulas in first-order logic:

- The foreign key from attribute buyer of relation order to relation buyer.
- The following functional dependency for relation product: $price, weight \rightarrow type$.
- No office products can cost more than $250 and weight less than 5 lb at the same time.
- Customers cannot order products that cost more then the limit of any of their credit cards.

Solution

$$\begin{aligned}
\mathbf{FK_1} :& \forall b, p, c : order(b, p, c) \rightarrow \exists x_1, x_2 : buyer(b, x_1, x_2) \\
\mathbf{FD} :& \forall n_1, t_1, p, w, n_2, w : product(n_1, t_2, p, w) \land product(n_2, t_2, p, w) \rightarrow t_1 = t_2 \\
\mathbf{light_expensive} :& \forall n, t, p, w : product(n, t, p, w) \rightarrow p \leq 250 \lor w \geq 5 \\
\mathbf{less_than_limit} :& \forall b_1, b_2, b_3, o_1, o_2, c_1, c_2, p_1, p_2, p_3 : buyer(b_1, b_2, b_3) \\
& \land order(b_1, o_1, o_2) \\
& \land card(c_1, b_!, c_2) \\
& \land product(o_1, p_1, p_2, p_3) \\
& \quad \rightarrow c_2 \leq p_2
\end{aligned}$$

Question 1.2.2 Creating Denial Constraints (13 Points)

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

- Products of category *transport* have to weight more than 1000 lb
- Translate the functional dependency $cardNum \rightarrow owner, limit$ on relation *card* into one or more denial constraints
- Buyers cannot place two orders for the same product with different counts.

Solution

$$d_1 : \forall n, t, p, w : \neg\,(product(n, t, p, w) \wedge t = transport \wedge w \leq 1000)$$

$$d_{21} : \forall c, o_1, l_1, o_2, l_2 : \neg\,(card(c, o_1, l_1) \wedge card(c, o_2, l_2) \wedge o_1 \neq o_2)$$
$$d_{22} : \forall c, o_1, l_1, o_2, l_2 : \neg\,(card(c, o_1, l_1) \wedge card(c, o_2, l_2) \wedge o_1 \neq o_2)$$

$$d_3 : \forall n, p, c_1, c_2, x_1, x_2 : \neg\,(buyer(n, x_1, x_2) \wedge order(n, p, c_1) \wedge order(n, p, c_2) \wedge c_1 \neq c_2)$$

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. Otherwise, state explicitly that no containment mapping exists.

Q_1`(X) :- R(X,X),S(X,Y),R(Y,X).`
Q_2`(C) :- R(A,B),R(B,C),R(A,A),S(C,A).`
Q_3`(Y) :- R(A,B),R(B,Y),S(Y,A).`
Q_4`(A) :- R(B,D),S(A,B).`

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

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

$\underline{Q_2 \to Q_3}$:
no containment mapping exists

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

$\underline{Q_3 \to Q_1}$:

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

$\underline{Q_3 \to Q_2}$:

$Y \to C$
$A \to A$
$B \to B$

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

$\underline{Q_4 \to Q_1}$:

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

$\underline{Q_4 \to Q_2}$:

$D \to A$
$A \to C$
$B \to A$

$\underline{Q_4 \to Q_3}$:

$D \to B$
$A \to Y$
$B \to A$