13/1/2020 Úvod do databáz, skúškový test, max **60** bodov

1. Uvažujte databázu bez duplikátov a null hodnôt: capuje(Krcma, Alkohol), lubi(Pijan, Alkohol), navstivil(Idn, Pijan, Krcma), vypil(Idn, Alkohol, Mnozstvo).
Platí: Idn $\rightarrow$ Pijan, Krcma; Idn, Alkohol $\rightarrow$ Mnozstvo; Mnozstvo > 0.
a) Sformulujte bezpečný dotaz v Datalogu **(6)** a relačnej algebre **(6)** na dvojice [P, A] také, že pijan P ľúbi alkohol A, a zároveň P vypil A v každej krčme, ktorá čapuje aspoň 3 alkoholy, ktoré P ľúbi.

Datalog:

```
answer(P, A) ←
        lubi(P, A),
        not niekde_nevypil(P, A).

niekde_nevypil(P, A) ←
        lubi(P, A), /* safety */
        lubi(P, A1),
        lubi(P, A2),
        lubi(P, A3),
        capuje(K, A1),
        capuje(K, A2),
        capuje(K, A3),
        not A1 = A2,
        not A1 = A3,
        not A2 = A3,
        not niekedy_vypil(P, A, K).

niekedy_vypil(P, A, K) ←
        navstivil(I, P, K),
        vypil(I, A, _).
```

Relačná algebra:

niekedy_vypil = $\pi_{\text{Pijan, Alkohol, Krcma}}$ (navstivil $\bowtie$ vypil)

niekde_nevypil =

$\pi_{\text{l.Pijan, l.Alkohol}}$

(

$\pi_{\text{l.Pijan, l.Alkohol, c1.Krcma}}$

(

P_{l} (lubi) $\bowtie_{\text{l.Pijan = l1.Pijan}}$

P_{l1} (lubi) $\bowtie_{\text{l.Pijan = l2.Pijan} \wedge \text{l1.Alkohol} \neq \text{l2.Alkohol}}$

P_{l2} (lubi) $\bowtie_{\text{l.Pijan = l3.Pijan} \wedge \text{l1.Alkohol} \neq \text{l3.Alkohol} \wedge \text{l2.Alkohol} \neq \text{l3.Alkohol}}$

P_{l3} (lubi) $\bowtie_{\text{l1.Alkohol = c1.Alkohol}}$

P_{c1} (capuje) $\bowtie_{\text{l2.Alkohol = c2.Alkohol} \wedge \text{c1.Krcma = c2.Krcma}}$

P_{c2} (capuje) $\bowtie_{\text{l3.Alkohol = c3.Alkohol} \wedge \text{c2.Krcma = c3.Krcma}}$

P_{c3} (capuje)

)

– niekedy_vypil

)

answer = lubi – niekde_nevypil

b) Sformulujte bezpečný dotaz v Datalogu **(6)** a SQL **(6)** na dvojice alkoholov [A1, A2] také, že A1 bol v každej krčme vypitý vo väčšom celkovom množstve než A2 (predpokladajte, že každá krčma čapuje nejaký alkohol a každý alkohol je niekde čapovaný).

Datalog:

```
answer(A1, A2) ←
        capuje(_, A1),
        capuje(_, A2),
        not zla_krcma(A1, A2).

zla_krcma(A1, A2) ←
        total(A1, K, T1),
        total(A2, K, T2),
        not T1 > T2.

total(A, K, T) ←
        subtotal(vpn(_, A, K, M), [A, K], [T = sum(M)]).
total(A, K, 0) ←
        capuje(_, A),
        capuje(K, _),
        not vpn(A, K).

vpn(I, A, K, M) ←
        navstivil(I, P, K),
        vypil(I, A, M).
vpn(A, K) ←
        vpn(_, A, K, _).
```

SQL:

```
with total as
(
	(
	select n.Krcma, v.Alkohol, sum(v.Mnozstvo) as T
	from navstivil n, vypil v
	where n.Idn = v.Idn
	group by n.Krcma, v.Alkohol
	)
	union
	(
	select c1.Krcma, c2.Alkohol, 0 as T
	from capuje c1, capuje c2
	where not exists
		(
		select *
		from navstivil n, vypil v
		where n.Idn = v.Idn and c1.Krcma = n.Krcma and c2.Alkohol = v.Alkohol
		)

	)
)
select c1.Alkohol, c2.Alkohol
from capuje c1, capuje c2
where not exists
	(
	select *
	from total1, total2
	where t1.Krcma = t2.Krcma and t1.Alkohol = c1.Alkohol and t2.Alkohol = c2.Alkohol and t1.T <= t2.T
	)
```

2. Uvažujte reláciu r(A, B, C, D, E) s funkčnými závislosťami
$A \rightarrow CE$, $ACD \rightarrow BE$, $BC \rightarrow D$, $BE \rightarrow AC$.
a) Uveďte príklad dekompozície danej relačnej schémy, ktorá je v Boyce-Coddovej normálnej forme a nie je bezstratová (nespája sa bezstratovo). Vysvetlite. **(6)**

Napríklad r1(A, C, E) a r2(B, D) je dekompozícia s požadovanými vlastnosťami.
Obe r1 aj r2 sú v BCNF. V r1 platia netriviálne závislosti $A \rightarrow C$, $A \rightarrow E$, lenže A je nadkľúč v r1. Iné „krátke“ netriviálne závislosti v r1 neplatia. r2 je v BCNF, lebo r2 je binárna relácia, v ktorej neplatí žiadna funkčná závislosť s prázdnou ľavou stranou.
Keďže $r1 \cap r2 = \{\}$ nie je nadkľúč v r1 ani v r2, uvedená dekompozícia sa nespája bezstratovo.

b) Rozhodnite či dekompozícia danej relačnej schémy do (A, B, D, E), (B, C, D) je v tretej normálnej forme. Odpoveď ÁNO resp. NIE zdôvodnite. **(6)**

Nájdime všetky kľúče r:
ABCDE

- -A: BCDE
  - -B: CDE
  - +B: BCDE
    - -C: BDE
      - -D: BE
        - -E: B
        - +E: **BE**
      - +D: BD
    - +C: BC
- +A: ABD
  - -B: **AD**
  - +B: **AB**

Kĺúče v r sú AB, AD, BE. Iné kľúče nie sú.

Jediný nekľúčový atribút je C.
Relácia (A, B, D, E) neobsahuje C, takže je v 3NF.
V (B, C, D) neplatí žiadna netriviálna funkčná závislosť s atribútom C na pravej strane. Teda aj táto relácia je v 3NF.

Áno, daná dekompozícia je v 3NF.

3. Uvažujte program s extenzionálnou databázou a(., .):
p(X, Y, Z) ← a(X, Y), a(Y, Z).
p(X, Y, Z) ← p(X, U, Y), p(U, Y, Z), not a(X, X), not a(Z, Z).
a) Vypočítajte výsledok dotazu ?- p(3, Y, Z) pre
a(., .) = {[0, 0], [0, 1], [1, 2], [2, 0], [2, 1], [2, 3], [3, 1]}. **(6)**

Použijeme naivnú evaluáciu, v ktorej sa iteruje

$p(., ., .) = \pi_{a1.1,\ a1.2,\ a2.2}\ (P_{a1}\ (a) \bowtie_{a1.2\ =\ a2.1} P_{a2}\ (a)) \cup$

$\pi_{p1.1,\ p2.2,\ p2.3}\ ((P_{p1}\ (p) \bowtie_{p1.2\ =\ p2.1} P_{p2}\ (p)) \rhd_{p1.1\ =\ a4.1\ \wedge\ p1.1\ =\ a4.2} P_{a3}\ (a) \rhd_{p2.3\ =\ a4.1\ \wedge\ p2.3\ =\ a4.1} P_{a4}\ (a))$

Inicializácia:
p(., ., .) = {}

Iterácia 1:
p(., ., .) = {[0, 0, 0], [0, 0, 1], [0, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 3], [2, 0, 0], [2, 0, 1], [2, 1, 2], [2, 3, 1], [3, 1, 2]}

Iterácia 2:
p(., ., .) = {{[0, 0, 0], [0, 0, 1], [0, 1, 2], [1, 0, 1], [1, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 3], [1, 3, 1], [2, 0, 0], [2, 0, 1], [2, 1, 2], [2, 2, 1], [2, 2, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1], [3, 2, 3]}

Iterácia 3:
p(., ., .) = {{[0, 0, 0], [0, 0, 1], [0, 1, 2], [1, 0, 1], [1, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 3], [1, 3, 1], [2, 0, 0], [2, 0, 1], [2, 1, 2], [2, 2, 1], [2, 2, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1], [3, 2, 3], [3, 3, 1]}

Iterácia 4:
p(., ., .) = {{[0, 0, 0], [0, 0, 1], [0, 1, 2], [1, 0, 1], [1, 1, 2], [1, 2, 0], [1, 2, 1], [1, 2, 3], [1, 3, 1], [2, 0, 0], [2, 0, 1], [2, 1, 2], [2, 2, 1], [2, 2, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1], [3, 2, 3], [3, 3, 1]}

Po dosiahnutí pevného bodu (keď sa obsah relácie p po iterácii nemení) aplikujeme selekciu a projekciu danú dotazom.
Výsledok dotazu ?- p(3, Y, Z) je .(Y, Z) = **{[1, 2], [2, 1], [2, 3], [3, 1]}.**

b) Rozhodnite či výpočet daného programu naivnou evaluáciou skončí pre každé naplnenie relácie a(., .) a pre každý dotaz. Zdôvodnite. **(6)**

Program je stratifikovaný. Relácii e(., .) priradíme stratum 1, relácii p(., ., .) priradíme stratum 2.
Áno, program skončí pre každú (konečnú) populáciu a(., .) a ľubovoľný dotaz.

4. a) Vysvetlite metódu časových pečiatok pre izoláciu transakcií. (Uveďte čo si systém pamätá, popíšte kedy a aké akcie robí scheduler. Akcie schedulera zapíšte v pseudokóde.) **(6)**

Každá transakcia T dostane pri štarte časovú pečiatku TS(T).
Každý objekt X má dve časové pečiatky, TSR(X) a TSW(X). Obe sú inicializované na $-\infty$.

Pred vykonaním $r_T(X)$ vykoná scheduler toto:

```
if (TS(T) < TSW(X))
        abort T;
else
        TSR(X) := TS(T);
```

Pred vykonaním $w_T(X)$ vykoná scheduler toto:

```
if ((TS(T) < TSR(X)) ∨ (TS(T) < TSW(X)))
        abort T;
else
        TSW(X) := TS(T);
```

b) Uveďte príklad rozvrhu dvoch transakcií, ktorý je obnoviteľný, dá sa generovať schedulerom, ktorý používa dvojfázové zamykanie a nedá sa generovať schedulerom, ktorý používa metódu časových pečiatok. Vysvetlite. **(6)**

Rozvrh **s1, s2, r2(X), c2, w1(X), c1** má požadované vlastnosti.

Tento rozvrh je obnoviteľný, lebo neobsahuje dirty read.
Dá sa generovať schedulerom, ktorý používa 2PL, takto:
s1, s2, rl2(X), r2(X), ul2(X), c2, wl1(X), w1(X), ul1(X), c1

Scheduler, ktorý používa metódu časových pečiatok, urobí pre tento vstupný rozvrh nasledujúce akcie:

s1	vyrobí a zapamätá časovú pečiatku TS(T1)
s2	vyrobí a zapamätá časovú pečiatku TS(T2)
r2(X)	TSR(X) := TS(T2)
c2	commituje T2 (a zahodí TS(T2))
w1(X)	abortuje T1 (a zahodí TS(T1)), lebo TS(T1) < TSR(X)