4/2/2020 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 → Pijan, Krcma; Idn, Alkohol → Mnozstvo; Mnozstvo > 0.
a) Sformulujte bezpečný dotaz v Datalogu **(6)** a relačnej algebre **(6)** na pijanov, ktorí ľúbia niektorý z alkoholov, ktoré sa čapujú v každej krčme; a buď ten alkohol vypili pri každej svojej návšteve krčmy alebo pri žiadnej.

Datalog:

```
answer(P) ←
        lubi(P, A),
        not niekde_necapuju(A),
        not niekedy_vypil(P, A).

answer(P) ←
        lubi(P, A),
        not niekde_necapuju(A),
        not niekedy_nevypil(P, A).

niekde_necapuju(A) ←
        lubi(_, A), /* safety */
        capuje(K, _), /* safety */
        not capuje(K, A).

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

niekedy_nevypil(P, A) ←
        lubi(P, A), /* safety */
        navstivil(I, P, _),
        not v(I, A).

v(I, A) ←
        vypil(I, A, _).
```

Relačná algebra:

niekde_necapuju = $\pi_{Alkohol}$ (($\pi_{Alkohol}$ (lubi) × π_{Krcma} (capuje)) − capuje)

niekedy_vypil = $\pi_{Pijan,\ Alkohol}$ (navstivil ⋈ vypil)

niekedy_nevypil = $\pi_{Pijan,\ Alkohol}$ ($\pi_{Idn,\ Pijan,\ Alkohol}$ (lubi ⋈ navstivil) ▷ $\pi_{Idn,\ Alkohol}$ (vypil))

/* answer */

π_{Pijan} ((lubi ▷ niekde_necapuju) ▷ niekedy_vypil) ∪ π_{Pijan} ((lubi ▷ niekde_necapuju) ▷ niekedy_nevypil)

b) Sformulujte bezpečný dotaz v Datalogu **(6)** a SQL **(6)** na dvojice [P, N], kde N je počet krčiem, v ktorých pijan P vypil viacej než polovicu z počtu svojich obľúbených alkoholov.

```
answer(P, N) ←
        subtotal(prepil_dost(P, K), [P], [N = count(K)]).

prepil_dost(P, K) ←
        subtotal(lubi(P, A), [P], [L = count(A)]),
        subtotal(nvl(P, K, A), [P, K], [V = count(A)]),
        2 * V >= L.

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

SQL:

```
with
lubi_pocet as
(
        select l.Pijan, count(distinct l.Alkohol) as L
        from lubi l
        group by l.Pijan
),
vypil_pocet as
(
        select n.Pijan, n.Krcma, count(distinct v.Alkohol) as V
        from navstivil n, lubi l, vypil v
        where n.Idn = v.Idn and n.Pijan = l.Pijan and l.Alkohol = v.Alkohol
        group by n.Pijan, n.Krcma
),
prepil_dost as
(
        select lp.Pijan, vp.Krcma
        from lubi_pocet lp, vypil_pocet vp
        where lp.Pijan = vp.Pijan and 2 * vp.V >= lp.L
)
select pd.Pijan, count(distinct pd.Krcma) as N
from prepil_dost pd
group by pd.Pijan
```

2. Uvažujte nasledujúci SQL dotaz nad reláciou a(X, Y) bez duplikátov a NULLs:

```
with g as
(
        (select a1.X, a2.X as Y from a a1, a a2 where not exists
                (select * from a a3 where a3.X = a2.X and a3.Y = a1.Y)
        ) union
        (select a2.X as Y, a1.X from a a1, a a2 where not exists
                (select * from a a3 where a3.X = a2.X and a3.Y = a1.Y)
        )
)
select distinct a1.X, a2.X as Y from a a1, a a2 where not exists
(select * from g g1 where g1.X = a1.X and g1.Y = a2.X);
```

a) Zapíšte daný dotaz v relačnom kalkule. **(6)**

V definícii g sú operandy union rovnaké relácie, takže daný dotaz je ekvivalentný dotazu

```
with g as
(
        (select a2.X as Y, a1.X from a a1, a a2 where not exists
                (select * from a a3 where a3.X = a2.X and a3.Y = a1.Y)
        )
)
select distinct a1.X, a2.X as Y from a a1, a a2 where not exists
(select * from g g1 where g1.X = a1.X and g1.Y = a2.X);
```

V relačnom kalkule:

{[X1, X2]:
∃Y1 ∃Y2
a(X1, Y1) ∧ a(X2, Y2) ∧
¬ /* g(X1, X2) */
(
∃Y1 ∃Y2
a(X1, Y1) ∧ a(X2, Y2) ∧
¬ /* .(X2, Y1) */
a(X2, Y1)
)
}

b) Zapíšte výpočet daného dotazu v relačnej algebre. **(6)**

$g = (\pi_X (P_{a1(X, Y)} (a)) \times \pi_Y (P_{a2(Y, Z)} (a))) - P_{a3(Y, X)} (a)$
$answer = (\pi_X (P_{a1(X, Y)} (a)) \times \pi_Y (P_{a2(Y, Z)} (a))) - g$

c) Vypočítajte výsledok daného dotazu pre
a(X, Y) = {[1, 1], [1, 4], [2, 1], [2, 3], [2, 4], [3, 1], [4, 1], [4, 4]}. **(6)**

$g = (\pi_X (P_{a1(X, Y)} (a)) \times \pi_Y (P_{a2(Y, Z)} (a))) - P_{a3(Y, X)} (a) = \{[1, 3], [2, 1], [2, 3], [2, 4]\}$
$answer = (\pi_X (P_{a1(X, Y)} (a)) \times \pi_Y (P_{a2(Y, Z)} (a))) - g =$
{[1, 1], [1, 2], [1, 4], [2, 2], [3, 1], [3, 2], [3, 3], [3, 4], [4, 1], [4, 2], [4, 4]}

Výsledkom dotazu je relácia
.(X, Y) = {[1, 1], [1, 2], [1, 4], [2, 2], [3, 1], [3, 2], [3, 3], [3, 4], [4, 1], [4, 2], [4, 4]}.

3. Relácia p(Id, Cena, ...) má 4 000 000 záznamov. Uložená je na disku s blokmi veľkosti 4kB. V jednom bloku je uložených 10 záznamov relácie p. Stredná doba prenosu diskového bloku je 10 ms (=0,01 s). Pre vykonanie SQL dotazu
select Id, Cena from p order by Cena;
je rezervovaných 500 blokov v RAM (bloky v RAM a na disku sú rovnako veľké).
a) Vysvetlite organizáciu pamäte a priebeh výpočtu (na úrovni diskových prenosov) fyzického operátora merge-sort pri vykonávaní daného dotazu. Odhadnite čas vykonávania dotazu. **(6)**

Relácia p je uložená v 400 000 blokoch (= 4 000 000 / 10).

Vo fáze vytvárania behov sa do rezervovaných 500 blokov v RAM prečíta 500 blokov relácie p, utriedi sa a zapíše do jedného behu. Toto sa zopakuje celkovo 800-krát, kým sa nespracuje všetkých 400 000 blokov relácie p.
Výsledkom je 800 utriedených behov, každý má dĺžku 500 blokov.
Každý blok relácie p sa raz prečíta z disku: 400 000 diskových prenosov.
Každý blok relácie p sa raz zapíše na disk: 400 000 diskových prenosov.

V zlučovacej fáze sa 1 blok použije pre výstupný beh, ostatných 499 blokov pre vstup utriedených behov (jeden blok pre jeden beh).
Prečíta sa 499 * 500 blokov: 249 500 diskových prenosov.
Každý z nich sa zapíše na disk: 249 500 diskových prenosov.

Zlučovacia fáza sa raz zopakuje. 1 blok RAM sa použije pre výstupný beh, pre vstup každého z 302 zlučovaných behov sa použije 1 blok RAM.
Každý blok relácie p sa raz prečíta z disku: 400 000 diskových prenosov.
Každý blok relácie p sa raz zapíše na disk: 400 000 diskových prenosov.

Spolu sa urobí 2 * (400 000 + 249 500 + 400 000) = 2 099 000 diskových prenosov. **Odhad času výpočtu je 20 990 s, čo je zhruba 5,83 hodín.**

b) Odhadnite čas vykonávania dotazu, ak sa pre merge-sort použije 5000 blokov RAM (namiesto 500). **(6)**

Vo fáze vytvárania behov sa do rezervovaných 5000 blokov v RAM prečíta 5000 blokov relácie p, utriedi sa a zapíše do jedného behu. Toto sa zopakuje celkovo 80-krát, kým sa nespracuje všetkých 400 000 blokov relácie p.
Výsledkom je 80 utriedených behov, každý má dĺžku 5000 blokov.
Každý blok relácie p sa raz prečíta z disku: 400 000 diskových prenosov.
Každý blok relácie p sa raz zapíše na disk: 400 000 diskových prenosov.

80 behov sa zlúči do jedného v 1 prechode.
Každý blok relácie p sa raz prečíta z disku: 400 000 diskových prenosov.
Každý blok relácie p sa raz zapíše na disk: 400 000 diskových prenosov.

Spolu sa urobí 2 * (400 000 + 400 000) = 1 600 000 diskových prenosov. **Odhad času výpočtu je 16 000 s, čo je zhruba 4,44 hodín.**

c) Má za účelom urýchlenia výpočtu daného dotazu zmysel vytvoriť index pre reláciu p? Ak áno, navrhnite typ indexu a vysvetlite ako pomáha pri redukcii času vykonávania. Ak nie, vysvetlite prečo v tomto prípade index nepomáha. **(6)**

Áno, index pomáha. Ak je vytvorený **B-stromový alebo B$^+$-stromový index podľa atribútu Cena**, tak netreba použiť merge-sort. Keďže listy indexu (dátové bloky) sú spojené dvojitým linkovaným zoznamom, stačí raz nájsť najľavejší list a potom raz prejsť po smerníkoch zoznamu všetky listy zľava doprava. **Nezávisle od veľkosti p či indexu, v RAM stačí rezervovať niekoľko málo blokov (2 stačia).**

Nájdenie najľavejšieho listu v stromovom indexe vyžaduje niekoľko málo diskových prenosov, ktoré môžeme zanedbať.
Listov v indexe je nanajvýš toľko, koľko je blokov relácie p. Každý sa prečíta raz, a spolu s každým čítaním sa prečíta príslušný blok relácie p, na ktorý ukazuje dátový pointer v indexe (kvôli nájdeniu príslušnej hodnoty atribútu Id).
Každý list indexu sa raz prečíta z disku: max. 400 000 diskových prenosov.
Každý blok relácie p sa raz prečíta z disku: 400 000 diskových prenosov.
Každý výstupný blok sa raz zapíše na disk: max. 400 000 diskových prenosov.

To je spolu najviac 1 200 000 diskových prenosov. **Odhad času výpočtu je 12 000 s, čo je zhruba 3, 33 hodín.**

Ak index vytvorený nie je, jeho vytvorenie sa amortizuje ak sa tento dotaz sa počíta opakovane, a ak sa relácia p často nemení.

Špeciálne pre výpočet tohto dotazu je výhodný **B-stromový alebo B$^+$-stromový index na dvojicu atribútov [Cena, Id]**. Vtedy sú totiž všetky potrebné dáta v indexe, takže vôbec netreba čítať dáta z relácie p.
Každý list indexu sa raz prečíta z disku: max. 400 000 diskových prenosov.
Každý blok výslednej relácie sa raz zapíše na disk: max. 400 000 prenosov.

Spolu sa urobí najviac 800 000 diskových prenosov. **Odhad času výpočtu je 8 000 s, čo je zhruba 2,22 hodín.**