TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einsatz und Realisierung von Datenbanken* im SoSe25

Alice Rey, Maximilian Reif, Tobias Goetz (i3erdb@in.tum.de)
http://db.in.tum.de/teaching/ss25/impldb/

Blatt Nr. 12

Hinweise: Dies ist das letzte Übungsblatt. Bitte nutzten Sie die Gelegenheit, offene Fragen oder Unklarheiten zu den Übungsblättern in das letzte Tutorium mitzubringen.

Hausaufgabe 1

Berechnen Sie das Ergebnis des Joins zwischen den Relationen in der folgenden Abbildung:

R	
A	B
a_0	b_0
a_0	b_1
a_0	b_2
a_0	b_3
a_1	b_0
a_2	b_0
a_3	b_0

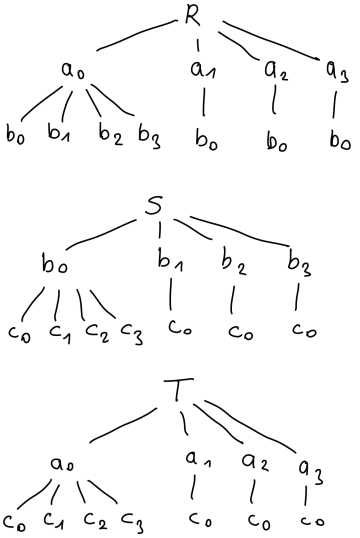
S	
B	C
b_0	c_0
b_0	c_1
b_0	c_2
b_0	c_3
b_1	c_0
b_2	c_0
b_3	c_0

T	
A	C
a_0	c_0
a_0	c_1
a_0	c_2
a_0	c_3
a_1	c_0
a_2	c_0
a_3	c_0

a) Berechnen Sie das Ergebnis des Joins mithilfe von zwei binären Joins (z.B. (R $\bowtie$ S) $\bowtie$ T). Wie viele Tupel enthält das Zwischenergebnis? Wie viele Tupel enthält das Endergebnis? **Lösung:** Das Zwischenergebnis (R $\bowtie$ S) enthält 19 Tupel. Das Endergebnis enthält 10 Tupel.

b) Was ist der Vorteil von Worst-Case Optimal Joins? Erklären Sie das Prinzip des Leapfrog Triejoin Algorithmus. **Lösung:** vgl. Buch

c) Bauen Sie aus den Relationen R, S und T Tries. **Lösung:**



d) Berechnen Sie erneut das Ergebnis des Joins, dieses Mal mithilfe des Leapfrog Triejoin Algorithmus. **Lösung:**

1. Gehe zu Knoten a_0 in Trie R und T.
2. Gehe zu Knoten b_0 in Trie R und S.
3. Vergleiche Kindknoten von Trie T(a_0) und S(b_0).
4. Erzeuge Ergebnis-Tupel (a_0, b_0, c_0), (a_0, b_0, c_1), (a_0, b_0, c_2,c_3).
5. Gehe zu Knoten b_1 in Trie R und S.
6. Vergleiche Kindknoten von Trie T(a_0) und S(b_1).
7. Erzeuge Ergebnis-Tuplel (a_0, b_1, c_0).
8. Für b_2 und b_3 wiederholen sich Schritte 5-7.
9. Gehe zu Knoten a_1 in Trie R und T.
10. Gehe zu Knoten b_0 in Trie R und S.
11. Vergleiche Kindknoten von Trie T(a_1) und S(b_0).
12. Erzeuge Ergebnis-Tuplel (a_1, b_0, c_0).
13. Für a_2 und a_3 wiederholen sich Schritte 9-12.

Hausaufgabe 2

Gegeben seien zwei dünnbesetzte Matrizen A und B.

$$A = \begin{pmatrix} 5 & 0 & 0 & 0 \\ 0 & 8 & 8 & 8 \\ 0 & 0 & 3 & 0 \\ 0 & 6 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 3 & 0 & 4 \\ 0 & 0 & 5 & 6 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Da dünnbestzte Matrizen nur sehr wenige Positionen haben mit Nicht-Null-Einträgen, können diese sehr effizient gespeichert werden. Statt die gesamte Matrix zu speichern, wird für jeden Nicht-Null-Eintrag die Spalte, die Zeile, sowie deren Wert gespeichert:

Matrix A

row	col	value
0	0	5
1	1	8
2	2	3
3	1	6

Matrix B

row	col	value
0	0	1
0	1	2
1	1	3
1	4	4
2	2	5
2	3	6

Formulieren Sie eine Anfrage in SQL, die das Produkt der Matrizen A und B (A×B) berechnet.

```
with sparse_matrix_A (row, col, value) as (
    SELECT *
    FROM (VALUES (0,0,5),(1,1,8),(2,2,3),(3,1,6))
),
sparse_matrix_B (row, col, value) as (
    SELECT *
    FROM (VALUES (0,0,1),(0,1,2),(1,1,3),(1,4,4),(2,2,5),(2,3,6))
)
```

```
select a.row, b.column, sum(a.value * b.value)
from sparse_matrix_A a, sparse_matrix_B b
where a.col = b.row
group by a.row, b.column
```

Hausaufgabe 3

Beantworten sie die folgenden Fragen in SQL. Das Schema ist aus dem TPC-H Benchmark. Schreiben sie je zwei Anfragen - eine mit und eine ohne Window-Funktionen.

1. Bestimmen Sie den Rang der Bestellungen aus der orders Relation nach dem Gesamtpreis (totalprice). Der höchste Gesamtpreis soll den Rang 1 erhalten.

```
select rank() over (order by o_totalprice desc),
    o_orderkey from orders;
-- without window function
select (select count(*) + 1
    from orders o2
    where o2.o_totalprice > o1.o_totalprice) rank,
    o_orderkey
from orders o1;
```

2. Berechnen sie für jede Bestellung die laufende Summe über den Gesamtpreis (totalprice) im jeweiligen Jahr (extract(year from o_orderdate)).

```
select sum(o_totalprice) over
(partition by extract(year from o_orderdate)
order by o_orderdate rows between unbounded preceding and current row)
from orders
--
select (select sum(o_totalprice)
from orders o2
where
extract(year from o2.o_orderdate)
= extract(year from o1.o_orderdate) and
and o2.o_orderdata <= o1.o_orderdate)
from orders o1
```