Database Systems WS 2023/24
Exercise 9: Distributed 08.01.2024, Due 15.01.2024 12:00 MEZ
Submitted by Erik Schwede, Suma Keerthi

1 Skyline Queries (1 P.)

Given the TPC-H[1] table **"part"**.

We want to compute the skyline over the following four dimensions:

- *p_size:* larger is better,
- *p_retailprice:* less is better,
- *p_container:* Use the function `getContainerSize` from below: less is better.
- *p_brand:* Cannot be compared.

In OLAT you will find a file `skylineFunc.sql`, which implements the `getContainerSize` function. You can either copy the create-statement from the file and execute it in your DBMS or you can directly execute the file with the following command (you may have to change the database name to the one you used to create it).

```
psql -d tpch -f skylineFunc.sql
```

1. Create a SQL query to calculate the number of elements in the skyline defined above (without using the `SKYLINE`-operator). Submit the query and the result.
2. Write the same query using the `SKYLINE`-operator. (You do not have to install the plugin and execute the query.)

2 Skyline NN (1 P.)

Implement the nearest neighbor (NN) method to compute skylines in a language of your choice. A basic Java template is available in OLAT. This template provides a (fake) R-Tree implementation which can be used to query for the nearest neighbor of a point, all points within a rectangle, and the nearest neighbor within a rectangle. If you are using another language, you are allowed to also implement a fake R-Tree with the methods described above (or use a real implementation without a included skyline function).

The program should output the calculated skyline (the template already takes care of this).

Submit your code and the output for the following points (If you use the template, the points are already included):

$$(10, 20), (12, 10), (8, 11), (16, 19), (6, 4), (5, 6), (14, 12), (2, 5), (3, 10), (13, 19), (17, 5), (9, 3), (20, 8), (8, 10)$$

[1] `http://dbis.informatik.uni-kl.de/files/teaching/ws1819/dbs/protected/tpch.dmp.gz`

Database Systems WS 2023/24
Exercise 9: Distributed 08.01.2024, Due 15.01.2024 12:00 MEZ
Submitted by Erik Schwede, Suma Keerthi

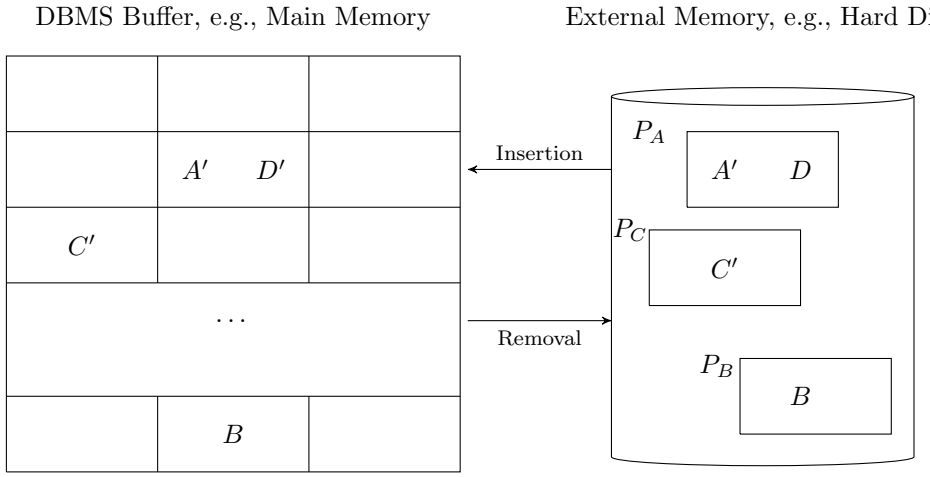
3 Transactions (1 P.)

1. A database has the consistency condition $0 \leq A \leq B$. Describe, for the following transactions, if the condition is fulfilled. Explain your answer.

   T_1 : `B:= 2*A; A:= 2*B;`
   T_2 : `B:= A+1; A:= B+1;`
   T_3 : `A:= A+2*B; B:= 2*A+B;`

2. For each of the previous transactions, provide the input, read, and write operations. Assume immediate write-back of the changed values. Describe the effect of these operations on the main memory and the hard disk. The initial values are $A = 5$ and $B = 10$.

3. Given two transactions T_1 and T_2 and the following situation (' marks changed entries):



The following table describes the operations of T_1 and T_2 at different times.

Time	T_1	T_2
0	READ(A,a)	READ(C,c)
10	a:=a+10	c:=c*2
20		READ(B,b)
30	WRITE(A,a)	
40	READ(D,d)	b:=b+c/4
50	d:=17*d+42	
60	OUTPUT(A)	WRITE(C,c)
70	WRITE(D,d)	OUTPUT(C)
80	OUTPUT(D)	WRITE(B,b)
90	COMMIT	
100		COMMIT

Discuss if the illustration matches the entries of the table. Does it only represent the table at a specific time (if yes, when and why) or does it not represent it at all?

If the system crashes during the execution, what would have to be done to guarantee ACID? Which parts of ACID are affected? How does the situation change at timestamp 101?

Submitted by Erik Schwede, Suma Keerthi

4 Recovery (1 P.)

In a DBMS three transactions T_1, T_2, and T_3 are executed concurrently. The data accessed performed by the transactions are stored in the log (Table 1).

1. The system crashes after step 19. None of the changes were written to the database, but the log is complete. Using this log, perform the three stages of recovery and explain what happens. Explain your steps in-detail.
2. Is the log changed after completing the recovery process? If yes, explain what changed.

	T_1	T_2	T_3	Log
				[LSN, TA, PageID, Redo, Undo, PrevLSN]
1.	BOT			$[\#01, T_1, -, BOT, -, 0]$
2.	$r(B, B_1)$			
3.	$B_1 = B_1 * 8$			
4.		BOT		$[\#02, T_2, -, BOT, -, 0]$
5.	$w(B, B_1)$			$[\#03, T_1, P_B, B = B * 8, B = B/8, \#01]$
6.		$r(A, A_1)$		
7.			BOT	$[\#04, T_3, -, BOT, -, 0]$
8.		$A_1 = A_1 + 9$		
9.		$w(A, A_1)$		$[\#05, T_2, P_A, A = A + 9, A = A - 9, \#02]$
10.	$abort$			$[\#06, T_1, -, abort, -, \#03]$
11.		$r(C, C_1)$		
12.			$r(A, A_2)$	
13.			$A_2 = A_2 + 3$	
14.		$C_1 = C_1/1$		
15.		$w(C, C_1)$		$[\#07, T_2, P_C, C = C/1, C = C * 1, \#05]$
16.		$commit$		$[\#08, T_2, -, commit, -, \#07]$
17.			$w(A, A_2)$	$[\#09, T_3, P_A, A = A + 3, A = A - 3, \#04]$
18.			$r(B, B_2)$	
19.			$B_2 = B_2/6$	
				Crash

Tabelle 1: Log