Ferhat Elmas 214805

Advanced Databases Hw-10

1-) Since we have 4 columns in cube query, we can have 15 different combinations(subset) of aggregates which makes in total 2^15-1=32267 - 1 partial materializations. However, we have 10000 tuples storage cost, so we consider combinations that have only 10000 tuples or less.

time	→	500
latitude	→	50
longitude	→	100
depth	→	20
time, depth	→	10000
latitude, longitude	→	5000
latitude, depth	→	1000
longitude, depth	→	2000

As it can be seen from above table, (time, depth) subset is bad because when we choose it, we hit the limit at start and we can choose any more subsets. However, all others can be chosen without problem, sum of others is less than 10000 rows.

Moreover, there is a significant difference between materializated query execution and naive scanning so we need to precompute aggregates as much as possible.

500 + 50 + 100 + 20 + 5000 + 1000 + 2000 = 8670 rows are needed to store these subsets.

As a result, we use all aggregates other than (time, depth) which are (time), (latitude), (longitude), (depth), (latitude, longitude), (latitude, depth), (longitude, depth).

2-)

Precedence Graph:

$$\begin{tikzcd}
 & T3 & \\
T1 \arrow[ur] \arrow[rr, shift left] & & T2 \arrow[ul] \arrow[ll, shift left]
\end{tikzcd}$$

It is serializable because data objects will have the same values after <T1, T2, T3> serial execution (without interleaved).

It is also view-serializable because this schedule has a view-equivalent to serial execution of <T1, T2, T3>.

- View equivalence comes from:
    - Read initial values must be same. In S, there is no initial value reading. No need to consider.
    - In S, T3 reads A written by T1 and it is also done in serial execution, and T3 reads B written by T2 in S, it is also done in serial execution.
    - Final values are written in S are also same in serial execution, A is written by T3 in S, so does T3 in serial execution and B is written by T2 in S, and T2 writes B in serial execution.

However, it isn't conflict-serializable because its precedence graph isn't acyclic and it contains a cycle between T1 and T2 of length 2.

Again, it isn't avoid-cascading-aborts because for example if T1 aborts, T3 should have been aborted since it reads a value of object A which is written by aborted transaction T1.