RWTH AACHEN UNIVERSITY

Implementation of Databases
Assignment #5

Due on January 10, 2016

Prof. Dr. rer. pol. Matthias Jarke

Dr. rer. nat. Christoph Quix

Submitted by:

Sanchit Alekh, Idil Esen Zulfikar

MSc. Software Systems Engineering

Problem 1

Query Optimization and Cost Estimation

1. Identify a relational algebra tree (or a relational algebra expression if you prefer) that reflects the order of operations a decent query optimizer would choose (applying the heuristics selection before join, and projections are done as many and as early as possible).

Solution

An efficient Relational Algebra tree is given in the **Figure 1**.

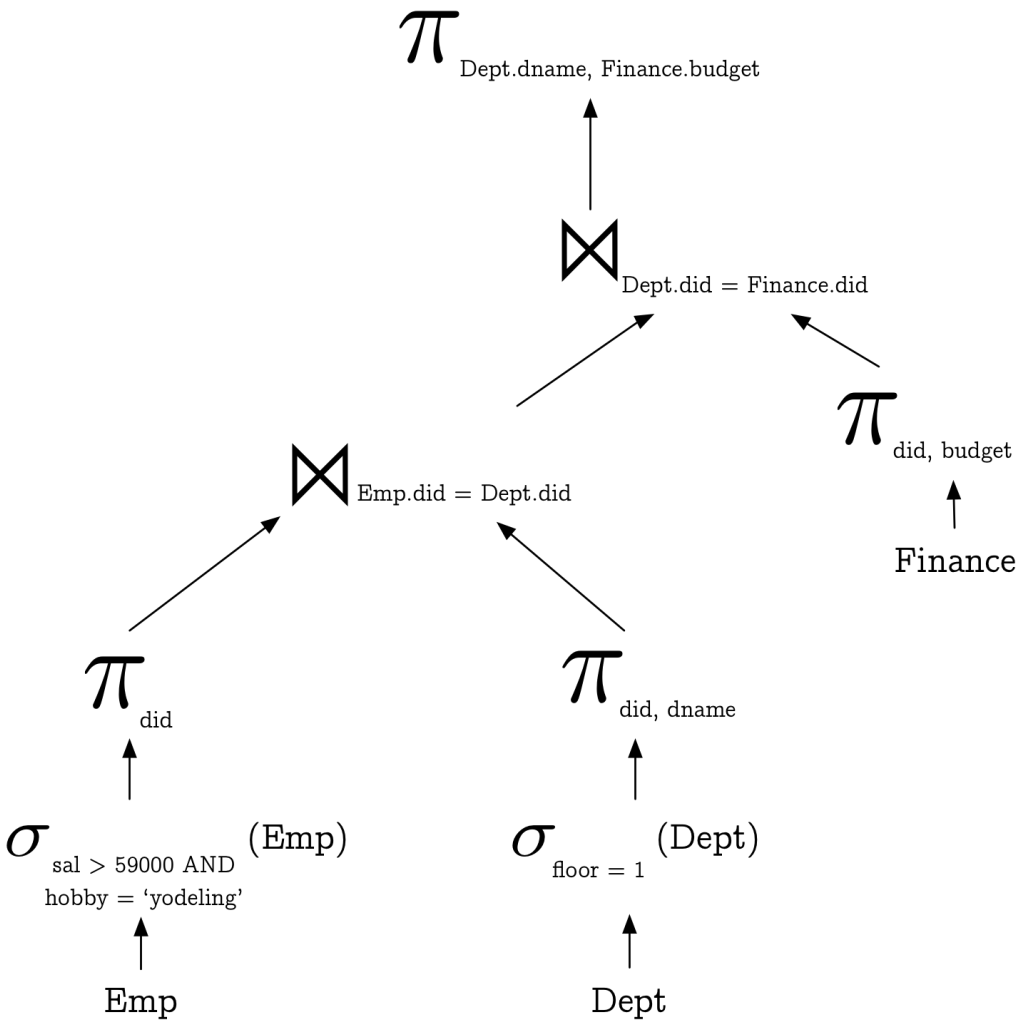

Figure 1: Relational Algebra Tree for Optimized Query Execution

2. Suppose that the following additional information is available: Unclustered B+ tree indexes exist on Emp.did, Emp.sal, Dept.floor, Dept.did, and Finance.did. Furthermore, assume that the costs for a tree traversal (from root to a leaf) are 3, additional costs for scanning leaves sequentially can be ignored. The systemâs statistics indicate that employee salaries range from 10.000 to 60.000, employees enjoy 200 different hobbies, and the company owns two floors in the building. All attribute values are uniformly distributed. There are a total of 50.000 employees and 5.000 departments (each with corresponding financial information, i.e., relation Finance has also 5.000 records) in the database. The DBMS used by the company has just one join method available: index nested loops.

   (a) For each of the query's base relations (Emp, Dept, and Finance) estimate the number of tuples that would be initially selected from that relation if all of the non-join predicates on that relation were applied to it before any join processing begins.

   ## Solution

   For **Emp** There are a total of 50000 employee tuples. There are two selection criteria, one on salary and the other on hobby.

   i. $\sigma_{salary>59000}$ Considering uniform distribution, there will be $\frac{50000}{50} = 1000$ employees earning salary between 59000 and 60000. Hence, 1000 tuples will be selected

   ii. $\sigma_{hobby=yodeling}$ Since the employees enjoy 200 hobbies, each hobby will have $\frac{50000}{200} = 25$ people for each hobby

   A conjunction between the two selection conditions will yield **a maximum of 25 tuples.**

   The selection on **Dept** is on the floor, i.e. $\sigma_{floor=1}$, which will yield $\frac{5000}{2} = 2500$ tuples.

   There is no selection criterion on **Finance**, so it will yield **5000** tuples.

   (b) Given your answer to the preceding question, what would be the total cost for a query plan that first performs the selections on Emp, then joins Emp with Dept, and then the result is joined with Finance. Assume again that selections are done before join, projections are done as many and as early as possible, and that the indexes are used if applicable. You can also assume, that intermediate results do not have to be stored (either, they fit into memory or they are pipelined to the next operator).

Solution

Note: For the corresponding solution, we assume that one page has a capacity to hold a maximum of 100 tuples

- There is an unclustered B+ Tree index on Emp.sal. Cost of $\sigma_{sal>59000}$ = Cost of traversing the B+ Tree + Retrieving each matched tuple from memory = $3 + 1000 = 1003$ I/Os.
- There is no index on hobby, so entire Emp relation has to be scanned. Emp has 50000 tuples, i.e. 500 pages. Cost of $\sigma_{hobby=yodeling} = 500$ I/Os
- There is an unclustered index on Dept.floor. Cost of $\sigma_{floor=1}$ = Cost of B+ Tree Access + Retrieval of matching tuples = $3 + 2500 = 2503$
- For Join of Dept and Emp, we need to calculate the number of pages in each relation left after selection. For Emp, it is $\lceil \frac{25}{100} \rceil = 1 page$. For Dept, it is $\lceil \frac{2500}{100} \rceil = 25$ pages
- Cost of $Emp \bowtie Dept$ using Index Nested Loop Join is $1+25\times1\times(3+1) = 101$ I/Os. Number of resulting tuples are $25 \times 2500 = 62500, or 625$ pages in the worst case
- Finance has $\frac{5000}{100} = 50 pages$. Cost of (Join Resultant of Emp and Dept) $\bowtie Finance = 50 + 100 \times 625 \times (3 + 1) = 250050$ I/Os

Therefore, the total cost involved in the query will be:
$1003 + 2503 + 500 + 101 + 250050 = 254157$ I/Os

Problem 2

Information Integration

Suppose we have a virtual data integration system with the following mediated schema:

(a) Provide a Local-As-View mapping between the two view predicates V1 and V2 and G.

Solution

$Times(Title, Time, Director) : -Movie(Title, Director, Genre) \wedge Schedule(Cinema, Title, Time)$
$Cinemas(Cinema, Genre, Title) : -Schedule(Cinema, Title, Time) \wedge Movie(Title, Director, Genre$

(b) Now suppose V1 and V2 are views defined over a database which consists of the following relations. Please do a rewriting of the following query against the view V1 such that you query the global schema using the given GAV mapping.

Solution

Query over V1 can be rewritten as:
$q(Title, Time, Director) : -V_1(\text{"}TheHobbit : TheBattleofFiveArmies\text{"}, \text{"23.12.2014"}, Director)$

Query can be rewritten over GAV mapping :
$q'(Title, Time, Director) : -Movies(\text{"}TheHobbit : TheBattleofFiveArmies\text{"}, Director) \wedge$
$Playing(\text{"}TheHobbit : TheBattleofFiveArmies\text{"}, \text{"23.12.2014"})$

Problem 3

Serialization and Recovery
Consider the following schedules:

(a) Write down the conflict relations $conf(s_n)$ of the schedules $s_n, n \in 1, 2, 3$.

Solution

$conf(s_1) = \{(w_2(y), r_3(y)), (w_3(x), r_1(x)), (w_3(x), w_2(x)), (r_1(x), w_2(x))\}$
$conf(s_2) = \{(w_2(y), r_3(y)), (r_1(x), w_2(x))\}$
$conf(s_3) = \{(w_2(y), w_3(y)), (w_2(y), r_3(y)), (r_1(x), w_2(x))\}$

(b) Determine for each schedule $s_n, n \in 1, 2, 3$ if it belongs to the classes CSR, OCSR and CO. Proof your decision.

Solution

In order to proof belonging to the classes CSR, OCSR and CO, the conflict graphs $G(s_n)$ are drawn.

If conflict graph $G(s_n)$ is acyclic, then it can be said the schedule s_n belongs to CSR class. As seen from Figure 2, $G(s_1)$ has two cycles that are $t_2t_3t_1t_2$ and $t_2t_3t_2$, so $G(s_1)$ is not acyclic and s_1 does not belong to CSR class. $OCSR \subset CSR$ and $CO \subset OCSR$ are valid, so s_1 also does not belong OCSR and CO.

$G(s_2)$ is acyclic, so s_2 belongs to CSR.
In order to proof CO, the following should be checked:
For all $t_i, t_j \in commit(s)$, $i \neq j$, with $(p, q) \in conf(s)$ for $p \in t_i$, $q \in t_j$, then: c_i is before c_j in s.

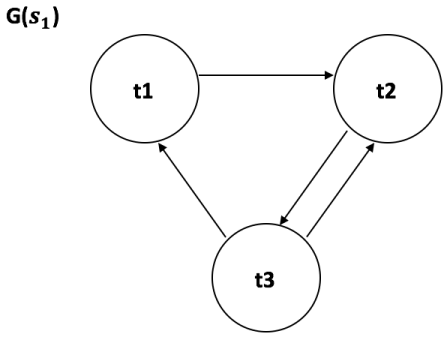

Figure 2: Conflict Graph of s_1

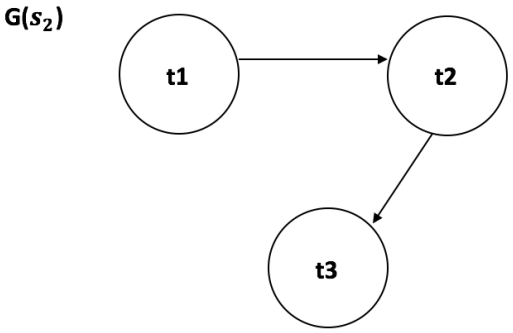

Figure 3: Conflict Graph of s_2

$conf(s_2)$ has two conflict relations that are between t_2 and t_3 and t_1 and t_2; moreover $t_1, t_2, t_3 \in commit(s_2)$. To provide the condition above, c_2 should be before c_3 and c_1 should be before c_2. In s_2, c_2 is before c_3 and c_1 is before c_2, so s_2 belongs to CO. Because of $CO \subset OCSR$, s_2 also belongs to OCSR.

$G(s_3)$ is acyclic, so s_3 belongs to CSR.
$conf(s_3)$ has three conflict relations that are between t_2 and t_3 and t_1 and t_2; moreover $t_1, t_2, t_3 \in commit(s_3)$. To provide belonging to the CO class, c_2 should be before c_3 and c_1 should be before c_2. However, c_3 is before c_2, so s_3 does not belong to CO.
Belonging OCSR can be checked with conflict serializable schedule. $s_3 \approx_c t_1 t_2 t_3$ means that this statement t_3 cannot occur completely before t_1,t_2 and t_2 cannot occur completely before t_1. There is not such a situation to violate the condition above, in the other words there is not any transaction that occurs completely before the other transactions. Therefore, s_3 belongs to OCSR.

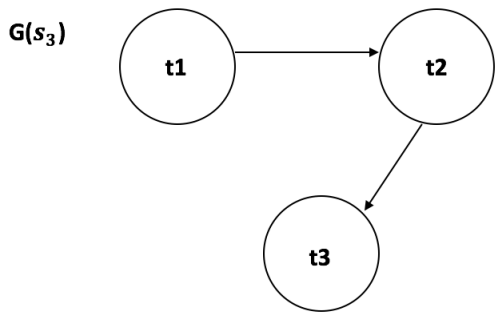

Figure 4: Conflict Graph of s_3

(c) Determine for each schedule $s_n, n \in 1, 2, 3$ if it belongs to the recovery classes RC, ACA and ST. Proof your decision.

Solution

Belonging RC recovery class is checked with the following condition:
$(\ \forall t_i, t_j \in trans(s), i \neq j)\ t_i$ reads from t_j in s $\wedge\ c_i \in s \implies c_j <_s c_i$
For s_1, t_1 reads x from $t_3 \wedge c_1 \in s_1 \implies c_3 <_s c_1$ and t_3 reads y from $t_2 \wedge c_3 \in s_1 \implies c_2 <_s c_3$ should be provided for belonging to RC. However, $c_1 <_s c_3$ and $c_3 <_s c_2$, so $s_1 \notin RC$.
For s_2, t_3 reads y from $t_2 \wedge c_3 \in s_2 \implies c_2 <_s c_3$ should be provided for belonging to RC. In the schedule, $c_2 <_s c_3$ and t_1 does not read x from the other transactions that means it does not violate any condition, so $s_2 \in RC$.
For s_3, t_3 reads y from $t_2 \wedge c_3 \in s_3 \implies c_2 <_s c_3$ should be provided for belonging to RC. However, $c_3 <_s c_2$ so $s_3 \notin RC$.

Belonging ACA recovery class is checked with the following condition:
$(\ \forall t_i, t_j \in trans(s), i \neq j)\ t_i$ reads x from t_j in s $\implies c_j <_s r_i(x)$
For s_1, t_1 reads x from t_3 in $s_1 \implies c_3 <_s r_1(x)$ and t_3 reads y from t_2 in $s_2 \implies c_2 <_s r_3(y)$ should be provided. However, $r_1(x) <_s c_3$ and $r_3(y) <_s c_2$, so $s_1 \notin ACA$.
For s_2, t_3 reads y from $t_2 \implies c_2 <_s r_3(y)$ should be provided for belonging to ACA. In the schedule, $c_2 <_s r_3(y)$ and t_1 does not read x from the other transactions that means it does not violate any condition, so $s_2 \in ACA$.
For s_3, t_3 reads y from $t_2 \implies c_2 <_s r_3(y)$ should be provided. However, $r_3(y) <_s c_2$ so $s_3 \notin ACA$.

Belonging RC recovery class is checked with the following condition:
$(\ \forall t_i, t_j \in trans(s))(\forall p_i(x) \in op(t_i), p \in \{r, w\})$
$w_j(x) <_s p_i(x), i \neq j \implies a_j <_s p_i(x) \vee c_j <_s p_i(x)$

For s_1, $w_3(x) <_s r_1(x) \implies a_3 <_s r_1(x) \vee c_3 <_s r_1(x)$ and $w_2(y) <_s r_3(y) \implies a_2 <_s r_3(y) \vee c_2 <_s r_3(y)$ should be provided. However, $r_1(x) <_s c_3$ and $r_3(y) <_s c_2$, so $s_1 \notin ST$.
For s_2, $w_2(y) <_s r_3(y) \implies a_2 <_s r_3(y) \vee c_2 <_s r_3(y)$ should be provided. In the schedule, $c_2 <_s r_3(y)$ and $w_3(z)$ & $w_2(x)$ do not affected the other transactions that means that means it does not violate any condition, so $s_2 \in ST$.
For s_3, $w_2(y) <_s r_3(y) \implies a_2 <_s r_3(y) \vee c_2 <_s r_3(y)$ should be provided. However, $r_3(y) <_s c_2$, so $s_3 \notin ST$.