BAN 610 Problem set 3 – Indexing, Query Optimizer, and Concurrency

Edit your submission in this word document, attaching the screenshots of the codes used for each question. Include narrative descriptions, outputs screenshot, or short answers when requested.

**Task 1**

Suppose we have the following two tables.

TA

|  |  |  |
| --- | --- | --- |
| NetID | Salary | Dept |
| 123 | 100000 | Mgmt |
| 321 | 200000 | Fin |
| 456 | 150000 | Ops |
| 678 | 120000 | Ops |
| 789 | 310000 | Fin |

DeptContact

|  |  |
| --- | --- |
| Dept | Phone |
| Mgmt | 123 |
| Ops | 1232 |
| Fin | 12314 |

Please implement the following relational algebra using SQL code

**SELECT \* FROM TA WHERE Salary > 150000;**

**SELECT NetID, Dept FROM TA;**

**SELECT NetID, Dept FROM (SELECT \* FROM TA WHERE Salary > 150000);**

**SELECT \* FROM**

**(SELECT \* FROM TA T INNER JOIN DeptContact DC**

**On T.Dept = DC.Dept) WHERE Dept in (“Fin”,”Ops”);**

**Task 2**

Suppose we have 100,000 records in a table, and each record takes up 50 bytes. We use a two-level sparse index to index this table and each index takes up 5 bytes. Suppose one block is capable of storing 500 bytes. Please answer (1) how many indexes are there in each level of index, (2) how many blocks each level of index takes up, and (3) how many blocks we need to search to locate one record for each level index.

|  |  |  |
| --- | --- | --- |
|  | How many indexes are there in the table? | How many blocks does each level of index takes up? |
| First level index | 100,000/10=10,000 indexes | =100,000/10 = 10,000 blocks |
| Second level index | 10,000/5=2,000 indexes | = 10,000/50 =  200 blocks |

For the First Index Level there are 10,000 blocks to search and a binary search can be perform in ceiling( = ceiling(13.288) = 14 blocks. For the Second Level, there are 200 blocks to search and a binary search can be performed in ceiling( = ceiling(7.644) = 8 blocks.

Therefore, with a two-level index, on average, 22 blocks must be searched in order to locate one record.

**Task 3**

Please store the keys 2,3,5,8,12,18,22,28 using closed addressing. Use the hash function: h(x)=x%10. Compete the following table. If multiple keys are stored with the same hash, use “->” to indicate linked list.

|  |  |
| --- | --- |
| Hash | Key |
| 0 | n/a |
| 1 | n/a |
| 2 | 2->12->22 |
| 3 | 3 |
| 4 | n/a |
| 5 | 5 |
| 6 | n/a |
| 7 | n/a |
| 8 | 8->18->28 |
| 9 | n/a |

**Task 4**

Store the keys 2,3,5,8,12,18,22,28 using linear probing. Use the hash function: h(x)=x%10. Complete the following table.

|  |  |
| --- | --- |
| Hash | Key |
| 0 | 28 |
| 1 | n/a |
| 2 | 2 |
| 3 | 3 |
| 4 | 12 |
| 5 | 5 |
| 6 | 22 |
| 7 | n/a |
| 8 | 8 |
| 9 | 18 |

**Task 5**

Store the keys 2,3,5,8,12,18,22,28 in an order 3 B tree. Please draw the tree, you can use any software or draw by hand. The correct answer is not unique.

A diagram of numbers and lines

Description automatically generated

**Task 6**

Suppose we have the following two tables.

TA

|  |  |  |
| --- | --- | --- |
| NetID | Salary | Dept |
| 123 | 100000 | Mgmt |
| 321 | 200000 | Fin |
| 456 | 150000 | Ops |
| 678 | 120000 | Ops |
| 789 | 310000 | Fin |

DeptContact

|  |  |
| --- | --- |
| Dept | Phone |
| Mgmt | 123 |
| Ops | 1232 |
| Fin | 12314 |

Please answer if the following two query plans return the same result. And if yes, which query plan runs faster? Please briefly explain your answer.

Plan 1:

Inner Join on each table then Predicate on Salary

Plan 2: )

Predicate on Salary then Inner Join with the other table

**Answer: Yes, both plans return the same result. The Second plan runs faster since it gets rid of values before executing the Inner Join. This likely makes the speed of joining faster than in Plan 1.**

**Task 7**

Suppose we have the following two tables.

TA

|  |  |  |
| --- | --- | --- |
| NetID | Salary | Dept |
| 123 | 100000 | Mgmt |
| 321 | 200000 | Fin |
| 456 | 150000 | Ops |
| 678 | 120000 | Ops |
| 789 | 310000 | Fin |

DeptContact

|  |  |
| --- | --- |
| Dept | Phone |
| Mgmt | 123 |
| Ops | 1232 |
| Fin | 12314 |

Please answer if the following two query plans return the same result. And if yes, which query plan runs faster? Please briefly explain your answer.

Plan 1:

Predicate on Salary then Projection to extract the respective NetIDs.

Plan 2: *(())*

Projection extracts relevant attributes, Predicate on Salary then a final Projection to extract the respective NetIDs.

**Answer: Yes, both plans return the same result. Plan 1 runs faster since it gets rid of values then takes the column that’s required. Plan 2 is slower because it wastes time executing a projection first then gets rid of values afterwords.**

**Task 8**

We have the following table.

TA

|  |  |  |
| --- | --- | --- |
| NetID | Salary | Dept |
| 123 | 100000 | Mgmt |
| 321 | 200000 | Fin |
| 456 | 150000 | Ops |
| 678 | 120000 | Ops |
| 789 | 310000 | Fin |

NetID takes 20 bytes, salary takes 6 bytes, Dept takes 10 bytes

Please calculate the following quantities:

**T(TA)=5**

**S(TA)=36 bytes**

**V(TA,NetID)=5**

**V(TA, Salary)=5**

**V(TA, Dept)=3**

**Task 9**

Please use the hash function h(x)=x%10 to store the keys: 1,12,22,10,8,7,31,42,35 using Robin Hood probing. Please also show the number of searches required to locate each key using the following format: (x,y), where x is the key, and y is the number of searches. Complete the following table.

|  |  |
| --- | --- |
| Hash | Key |
| 0 | 10 |
| 1 | 1 |
| 2 | 12 |
| 3 | 31 |
| 4 | 22 |
| 5 | 42 |
| 6 | 35 |
| 7 | 7 |
| 8 | 8 |
| 9 | n/a |

**Task 10**

Please draw the precedence plot for the following transaction schedule

S = w1(A)r2(A)r3(A)w4(A)r1(A)

Is this schedule serializable? If yes, what is the equivalent serial schedule?

**Since there contains a cycle in the precedence graph on the left, the schedule is not serializable.**

**Task 11**

Is each of the following schedule 2PL? If yes, is it serializable?

**Answer: 2PL? Yes.**

**Serializable? No.**

**Answer: 2PL? No, shared lock not permitted on resource with active X lock. Serializable? No.**

**Answer: 2PL? No, S lock cannot be upgraded to X lock if another transaction has an active X lock on the resource already.**

**Serializable? No.**

**Answer: 2PL? No, T1 tries to write without an X lock and another X lock was already aquired on the resource of interest.**

**Serializable? No.**

**Task 12**

We are doing inner join for three relations R1(A,B), R2(B,C), and R3(C,D).

And we know the following:

* For R1: T(R1) = 2000, V(R1,A)=50, V(R1,B)=100
* For R2: T(R2) = 3000, V(R2,B)=200, V(R2,C)=300
* For R3: T(R3) = 1000, V(R3,C)=90, V(R3,D)=500

Which two relations should we join first to minimize the peak memory consumption (i.e., the maximum total number of records needed to stay in the memory in any time to complete the inner join)?

R1 and R2 First

T(R1 x R2 = R’) = T(R1)\*T(R2) / Max(V(R1,B), V(R2,B)) = (2000 \* 3000) / 200 = 30,000

T(R’ x R3) = (30000 \* 1000) / Max(V(R’,C), V(R2,C)) = 30,000,000 / 300 = 100,000

I/O = 2000 + 3000 + 30000 + 1000 = 36,000

Time Complexity = 36,000,000

R2 and R3 First

T(R2 x R3 = R’) = T(R2)\*T(R3) / Max(V(R2,C), V(R3,C)) = (3000\*1000) / 300 = 10,000

T(R’ x R1) = (10,000 \*1000) / 200 = 50,000

I/O = 3000 + 1000 + 10,000 + 1000 = 15,000

Time Complexity = 13,000,000

**Answer: Joining R2 and R3 first would be more efficient both on I/O and Time Complexity.**