CS143 Winter 2020
MIDTERM EXAM Solutions

- Problem A. Searching and Joining— 30 points
- We store the following file in blocks having size 4096 bytes. **partSup(pid char(20), Supplier char(30))**

Please, answer the following three questions assuming that our relation has 2 millions tuples, which are stored unspanned. Also there is a sparse B+ index on pid and there is no other index: Here are the questions:

1. How many blocks are needed to store the whole relation?

Answer: Since 4096/50 = 81.92, we can store 81 unspanned tuples in each block. Thus $\lceil 2,000,000/81 \rceil = 24692$ blocks.

Assuming that the B+ tree has two levels, how many blocks must be retrieved to execute the following query:

SELECT Supplier FROM partSup WHERE pid =′ XBC98629450000000000′

Answer: 2 blocks of the B+ tree and one from the file, for a total of 3.

3. Consider now the following query that finds pairs ofparts supplied by the same supplier:

- SELECT PS1.pid, PS2.pid
  FROM partSup as PS1 PS2
  WHERE PS1.Supplier = PS2.Supplier and PS1.pid < PS2.pid
- What join method will the system use to support this query, and what will be its estimate cost in terms of blocks fetched from disk assuming that only three main-memory blocks can be used to compute the join?
- A**nswer**: The index on **pid** cannot be used to answer this query. Thus the system will use a block join. With only 3 blocks available we have the worst case situation where the cost for two tables with respectively m blocks and n blocks is $n + n \times m$. In this self-join we have n = m and once we load one block from the first file we do need to reload it from the second so $n+n\times(n-1) = n\times n$.
  Both 24692×24692 and $24692 + 24692 \times 24692$ will be graded as correct answers.

Problem B. B+ trees — 20 points

Same relation as in Problem A, with 2 millions tuples stored unspanned on blocks of size 4096 bytes We have a sparse index on pid organized as a B+ tree. Pointers take 10 bytes. Answer the following questions.

1. What is the minimum number of nodes needed for the B+ tree? Answer for best case:

$N - 1 = \lfloor (4096 - 10)/(20 + 10) \rfloor = 136$. Thus $N = 137$.
Leaf Level: 137-1=136 pointers to the file. Sparse index pointing at 24,692 blocks. Since 24692/136 = 181.55, i.e. 182 leaf nodes are used.
First Level: N=137 pointers are used. Thus, $\lceil 182/137 \rceil = 2$ blocks are used.
One root: thus the total number of nodes in the best case is 1 + 2 + 159 = 162.

2. What is the maximum number of nodes needed for the B+ tree, in the worst case scenario (i.e., the situation which requires most blocks). Answer for worst case:

We still have N = 137.
Leaf Level: Worst case $\lceil (137 + 1)/2 \rceil = \lceil 138/2 \rceil = 69$. Thus 68 pointers to the file in leaf nodes Sparse index pointing to 4,692 blocks. Since 24,692/68 = 363.11 we conclude that 363 blocks are used at the leaf level (Why not 364 ?).
First Level: $\lceil 137/2 \rceil = 69$ pointers in each node. Thus $\lfloor 363/69 \rfloor = \lfloor 5.29 \rfloor$ nodes are needed. (Why floor rather than ceiling–i.e., why 5 rather than 6?)
One root: thus the total of nodes in the worst case is 1 + 5 + 363 = 369.

Problem C. SQL — 20 Points

- Given the table: taken(StNo, CourseID, Year, Quarter, Sec, Grade, Remarks):
- write an SQL query to find the students who got a grade less than class average in 7 or more classes they took —a class is identified by (CourseID, Year, Quarter, Sec) and Grade in taken is of type numeric. In your answer, the depth of nesting of sub-queries should not exceed 2.
- ANSWER. We are seeking students who got a grade less than class average in 7 or more classes they took

```
select  StNO from taken as T1
  where  Grade  > (select avg(Grade) from taken as T2
                                        where T2.CourseID=T1.CourseID and
                                       T2.Year= T1.Year and
                                     T1.Quarter= T2.Quarter and
                                   T2.Sec= T1.Sec)
                                   )
group by T1.StNO having count(*) >=7
```

Problem D. Potpourri: 30 points (5 Points per question):

Please answer the following questions. You must give a YES—NO answer to questions D1–D4 and if your answer is YES you must also write the equivalent RA expression.

D1 Can the intersection of relations R(A,B) and S(A,B) be expressed using only natu- ral joins?

Answer: Yes. $R \cap S = R \bowtie S$.

D2 Can the intersection of relations R(A,B) and S(A,B) be expressed using the set difference operator?

Answer: Yes. R∩S=R−(R−S).

D3 Can the intersection of relations R(A,B) and S(A,B) be expressed using the cartesian product and projection operator

Answer: No.

D4 Can the intersection of relations R(A,B) and S(A,B) be expressed using the carte- sian product, selection and projection operators?

Answer: Yes.

$$R \cap S = \pi_{R.A,R.B}(\sigma_{R.A=S.A \wedge R.B=S.B}(R \times S)).$$

Problem D (cont.)

- **D5** A relation R is indexed on its candidate key using an extendible hashing: Is this index (i) dense, (ii) sparse or (iii) it could be either way? You must explain the reason for the answer you selected.

Answer: This must be a dense index. Sparse indexes only work if we can perform ordered searches as in B+ trees.

- **D6** For the extendible hashing on R described in D5: does the structure of its directory and the number of the buckets it uses depends on (i) the order in which the tuples in R have been inserted, or (ii) they only depend on the values those tuples? You must explain the reason for the answer you selected.
- Answer: The overall structure depends only on the set of values. Different insertion orders might affect the order in which tuples are arranged in the buckets, but not the overall content of the buckets, nor the directory.