CS3402 Database Systems

Homework 3

Question 1. (50 marks)

Suppose that you have formatted your disk with a block size of 2048 bytes and assume that we have 50,000 CAR records of fixed length. A block pointer is 5 bytes long (P=5), and a record pointer is 6 bytes long (Pr=6). Each CAR record has the following fields: Model (20 bytes), Registration (8 bytes), Color (9 bytes), License (9 bytes), Location (40 bytes), Mileage (10 bytes), Price (8 bytes), Year (4 bytes), Manufacturer (16 bytes), and Remarks (200 bytes). The file is ordered by the key field Registration and we want to construct a primary index on Registration.

(a) Calculate the blocking factor (bfr) and the number of file blocks needed to store the CAR records. Assume that records are stored unspanned. How much space remains unused per block? . [10 marks]

Solution: The record size R = 20 + 8 + 9 + 9 + 40 + 10 + 8 + 4 + 16 + 200 = 324. To calculate the blocking factor, we get bfr = floor(B / R) = 6 records/block. The number of blocks is I = ceiling(50,000 / bfr) = 8334 blocks. Therefore, the unused space per block is B – (R * bfr) = 104 bytes.

(b) Calculate the index blocking factor bfri . [10 marks]

Solution: Each index entry has size Ri = size(registration) + P = 8 + 5 = 13 bytes. The index blocking factor is bfri = floor(B / Ri) = 157 entries / block.

(c) Assume that the primary index is a single-level index. Calculate the number of index entries and the number of index blocks. [10 marks]

Solution: We need one index entry per file block, so to calculate the number of index and we already calculate the number of blocks needed to store all 50,000 file records in Q1, which is I=8334. Thus, the number of index blocks: Bi = ceil(I / bfri) = 54 blocks.

(d) Now suppose that we want to make the primary index a multilevel index. How many levels are needed and what is the total number of blocks required by the multilevel index? . [10 marks]

Solution: The fan-out for the multilevel index is the same as the index blocking factor (bfri) which is 157 (see above). The number of 1st level blocks L1 is already calculated in 2b, and so we have L1=54. The number of 2nd level blocks

is L2 = ceil(L1 / fo) = 1. The total number of index blocks is L1 + L2 = 54 + 1 = 55 blocks.

(e) Consider the multilevel index from subquestion (d). What is the number of block accesses needed to search for and retrieve a record from the file given its Registration value? . [10 marks]

Solution: This is given as the number of index levels + 1, which, according to c) is 2+1=3.

Question 2. [20 marks]
Suppose that the size of a search key field is V=9 bytes, the size of a record pointer is Pr=7 bytes, the size of a block pointer/tree pointer is P=6 bytes, the order of internal nodes (p) is 20, and the number of data record pointers for a leaf node 20 for a B+ tree. What should be the minimum required block size B for this B+ tree.

Solution:
For internal nodes:
$(p * P) + ((p - 1) * V) \leq B$
$(20 * 6) + ((20 - 1) * 9) \leq B$
$291 \leq B$ [10 marks]

For leaf nodes:
$(pleaf * (Pr + V)) + P \leq$
$B\ (20 * (7+ 9)) + 6 \leq B$
$326 \leq B$ [10 marks]

The minimum required block size B is 326.

Question 3. [30 marks]

Consider three transactions T1, T2, and T3 and two schedules S1, S2, which are given below:

T1: r1 (X); r1 (Z); w1 (X);

T2: r2 (Z); r2 (Y); w2 (Z); w2 (Y);

T3: r3 (X); r3 (Y); w3 (Y);

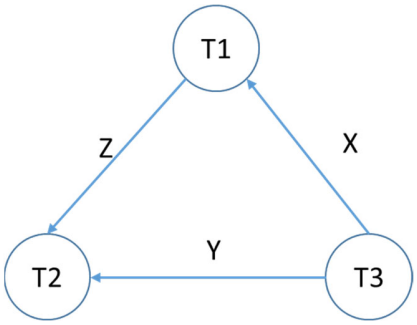
S1: r1(X); r2(Z); r1 (Z); r3(X); r3(Y); w1(X); c1; w3(Y); r2(Y); c3; w2(Z); w2(Y); c2;

S2: r1(X); r2 (Z); r3 (X); r1(Z); r2 (Y); r3 (Y); w1 (X); c1; w2 (Z); w3 (Y); w2 (Y); c3; c2;

1) Draw the serialization graphs for S1, S2 and state whether each schedule is serializable or not. If a schedule is serializable, write down all equivalent serial schedule(s). [20 marks]

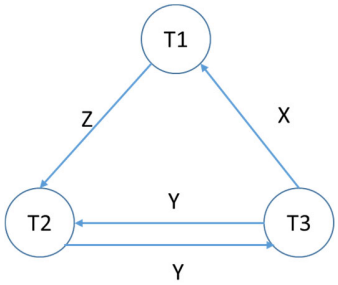
Solution:

S1:



It is serializable because the graph doesn't contain a cycle. The only equivalent serial schedule is T3 -> T1 -> T2.

S2:



The schedule isn't serializable because there is a cycle.

2) Please determine the strictest recoverability condition (non-recoverable, recoverable, cascadeless or strict) that each schedule satisfies.[10 marks]

Solution: S1 is recoverable since T2 reads Y which is written by T3, and T3 commits before T2 commits. But it is non-cascadeless, since when T2 reads Y, T3 has not committed yet. [5 marks] S1 is recoverable since T2 reads Y which is written by T3, and T3 commits after T2 commits. But it is non-cascadeless, since when T2 reads, T3 has not committed yet.

S2 is cascadeless, because every transaction reads only the items that are written by committed transactions. But it is not strict since T3 writes Y before T2 but when T2 writes Y T3 has not committed yet.