UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR11199 ADVANCED DATABASE SYSTEMS

Monday 23 rd May 2022

13:00 to 15:00

INSTRUCTIONS TO CANDIDATES

Answer any TWO of the three questions. If more than two questions are answered, only QUESTION 1 and QUESTION 2 will be marked.

All questions carry equal weight.

This is an OPEN BOOK examination.

MSc Courses

Convener: A.Pieris
External Examiners: A.Cali, V.Gutierrez Basulto.

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

CQ Evaluation and static analysis

1. (a) Consider the conjunctive queries

$$Q_1(x, y) :- R(y, x), R(z, x), R(v, x), R(x, u)$$

and

$$Q_2(x, x) :- R(x, x)$$

where x, y, z, u, v are variables.

Homomorphism theorem

State whether $Q_1 \subset Q_2$, $Q_2 \subset Q_1$, or $Q_1 \equiv Q_2$. Justify your answer. Recall that $Q_1 \subset Q_2$ means Q_1 is contained in Q_2, but Q_2 is not contained in Q_1. [*5 marks*]

Query optimisation

(b) Consider a database with relations `Users`, `Restaurants`, and `Reviews`.

Relation	Records	Record Size	Primary key
`Users(uid, uname, age)`	25,000	200 bytes	`uid`
`Restaurants(rid, rname, city)`	1,000	100 bytes	`rid`
`Reviews(uid, rid, feedback)`	20,000	400 bytes	`(uid,rid)`

`Reviews.uid` and `Reviews.rid` are foreign keys that reference `Users.uid` and respectively `Restaurants.rid`. Primary key attributes are underlined.

Consider the following SQL query that finds all reviews for the restaurants in Edinburgh made by users younger than 39.

```
SELECT *
  FROM Reviews JOIN Users ON Reviews.uid = Users.uid
  JOIN Restaurants ON Reviews.rid = Restaurants.rid
 WHERE Restaurants.city = 'Edinburgh' AND Users.age < 39
```

There are 20 different cities, uniformly distributed among the records of `Restaurants`. The distribution of `Users.age` has the following histogram:

15-24	25-34	35-44	45-54	≥ 55
17%	12%	15%	20%	36%

Assume the following:

- The page size is 1024 bytes.
- On each page, 1000 bytes can be used to store fixed-length records; the remaining 24 bytes are used for the page header.
- Each record is stored on one page only (i.e., records cannot span multiple pages).
- The buffer pool has 10 buffer pages.

QUESTION CONTINUES ON NEXT PAGE

i. What is the estimated number of records in the output of the above SQL query? Explain how you computed the answer. [*3 marks*]

ii. Consider query plans for the above SQL query that satisfy the following:

- The only join operator to be considered is sort-merge join, *without* the refinement combining sorting and merging, as discussed in class.
- No pipelining is to be used, that is, for each operator, including selection and join, the output is first written entirely to disk by one operator before it is read by the next operator.

From this set of plans, choose the optimal query plan in terms of the estimated number of I/O operations. Write down the optimal query plan as an operator tree. [*3 marks*]

iii. For each node of the chosen query plan, compute the size of its output in terms of the number of records and the number of pages, plus the cost incurred at that node. In addition, state the overall cost. Ignore the cost of writing the final result. Write down all the formulas you use. [*8 marks*]

Buffer management

(c) Why does the buffer manager use a replacement strategy and does not replace any page with pinCount = 0? [*2 marks*]

(d) Why do buffer frames in the buffer pool have a pin counter instead of just a pin flag? [*2 marks*]

Tree based indexing

(e) Why do database systems support B^+ tree indices instead of using existing binary search data structures such as binary search trees, Red-Black trees, and AVL trees? [*2 marks*]

Database systems favor B+ tree indices over binary search trees, Red-Black trees, or AVL trees primarily due to their disk I/O efficiency and ability to store large amounts of data. B+ trees have "high fan-out", which reduces the number of levels and, consequently, the number of disk accesses required during search operations. This structure is particularly suited for storage systems where reading from or writing to disk is much slower than operations in memory.

Additionally, B+ trees keep data sorted and allow efficient range queries by linking leaf nodes, further enhancing retrieval operations, essential for performance in database systems.

d) Buffer frames in the buffer pool use a pin counter instead of just a pin flag to accommodate scenarios where multiple processes or transactions simultaneously require access to the same page.
Each process "pins" the page by incrementing the counter, indicating its active usage. This mechanism ensures accurate tracking of how many operations are depending on a particular page at any given time. Once all operations are completed, each will decrement the counter. When the pin counter returns to zero, it signifies that no processes are using the page, making it eligible for eviction if needed. This system prevents premature page removal and maintains data integrity.
Eg: T1 sets F1 to True, T2 sets F1 to True, T2 sets F2 to False when it is over but T1 is not over yet. Eviction of this page due to a false flag would cause data integrity issues.

CQ Fast Evaluation

2. (a) Consider the Boolean conjunctive query

$$Q :- R(x, y, z), R(z, v, w), S(x, u), S(w, u)$$

where x, y, z, u, v, w are variables.

i. Give the hypergraph $H(Q)$ of the conjunctive query Q. *[2 marks]*

ii. Is $H(Q)$ acyclic? If yes, then give a join tree of $H(Q)$; otherwise, prove that $H(Q)$ is indeed not acyclic. *[3 marks]*

Transactions

(b) Consider the following schedule S over transactions $T_1 - T_7$:

$$S : \langle W_1(B), R_1(C), R_1(E), R_2(A), R_2(D), W_3(E), R_4(E), W_4(B), R_5(B), R_5(C), R_6(D), W_6(A), R_7(B), W_7(C), R_7(D) \rangle$$

Here, $W_i(A)$ represents a write operation on object A by transaction T_i, while $R_i(A)$ represents a read operation on object A by transaction T_i.

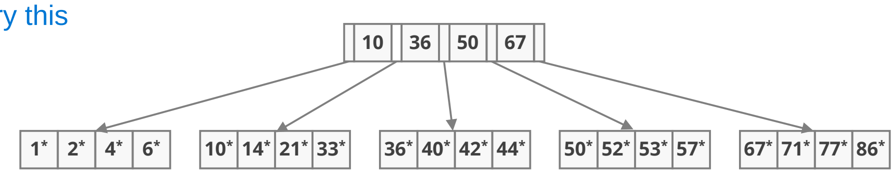
i. Give the *conflict graph* for S and use it to determine if the schedule is *conflict-serializable*. State your reasoning explicitly. *[3 marks]*

ii. If the schedule is conflict-serializable, how many conflict-equivalent serial schedules are there? Briefly describe (in 2-3 sentences) a high level algorithm that constructs all such serial schedules. No need to write down all the serial schedules.
If the schedule is *not* conflict-serializable, identify a minimal set of operations that need to be removed from S to make it conflict-serializable, and give one conflict-equivalent serial schedules after the removal. *[4 marks]*

Tree based indexing

(c) Draw the final B$^+$ tree that results from inserting index entries 29*, 51*, 37*, 45*, 58*, 38*, 80* in sequence into the following B$^+$ tree.

Try this



Where allowed use *redistribution* before *splitting*. When splitting, the new right node gets the larger number of entries.

Show the intermediate B$^+$ trees that led to the final B$^+$ tree. *[6 marks]*

QUESTION CONTINUES ON NEXT PAGE

Tree based indexing

(d) Let $\mathcal{B}$ be a *minimally filled* B^+ tree of degree d and height $h > 0$. Every inner node as well as the root of a minimally filled B^+ tree contains the minimal number of separator keys. Every leaf node of such a tree contains the minimal number of entries.

i. How many leaf node entries does $\mathcal{B}$ contain? Provide a closed formula over d and h. Justify your answer. 2 * (d + 1)^h-1 * d [*3 marks*]

ii. What is the height of $\mathcal{B}$ after deleting 1 entry? Justify your answer. [*2 marks*]

h-1

iii. What are the possible values for the height of $\mathcal{B}$ after deleting any d entries? Justify your answer. [*2 marks*]

3. (a) A database system implementing the ARIES recovery protocol has just restarted from a crash. The log below was found on disk. All pages were flushed to disk at and including LSN 50, so the log record has been truncated to start at LSN 60. Assume that no page has been flushed to disk after LSN 50, that is, all pages on disk have pageLSN < 60.

LSN	Trans ID	Type	PageId	prevLSN
		...		
60	T1	BEGIN-TXN		nil
70	T1	UPDATE	P2	60
80	T2	UPDATE	P3	40
90		BEGIN-CHECKPOINT		
100		END-CHECKPOINT		
110	T2	UPDATE	P1	80
120	T1	UPDATE	P3	70
130	T3	BEGIN-TXN		nil
140	T3	UPDATE	P1	130
150	T1	ABORT		120
160	T1	CLR LSN 120, undoNextLSN: 70	P3	150
170	T2	COMMIT		110
		*** CRASH ***		

i. Show the active transaction table and the dirty page table as recorded in the end checkpoint record. [*3 marks*]

ii. Show the active transaction table and the dirty page table for the same log after the analysis phase. [*3 marks*]

iii. Show all the new records added to the log after the UNDO phase is complete. [*4 marks*]

Distributed transactions

(b) Consider a database consisting of relations $R(A, B)$ and $S(B, C)$. Relation R consists of 600 pages, while S consists of 900 pages. Each page is 2KB.

Suppose we have three machines available to join R and S on $R.B = S.B$. Each machine has 101 buffer pages to perform the join. Ignore the cost of writing the final join result back to disk.

QUESTION CONTINUES ON NEXT PAGE

i. If both relations are initially round-robin partitioned across the machines, what is the expected *total amount of data in KB* transmitted over the network when computing parallel hash join? Explain your reasoning. [*2 marks*]

ii. Suppose we can combine the partitioning phase and the first phase of hashing, thus avoiding writing and reading intermediate data on disk between the two phases on each machine.
If both relations are initially round-robin partitioned across the machines, how many disk I/Os are needed in *total across all machines* to perform parallel hash join? Explain your reasoning. [*4 marks*]

iii. Suppose we evenly range-partition R and S on the B attribute across all machines, using the same range of values for R and S on each machine. How many disk I/Os are needed *total across all machines* to perform parallel sort-merge join? Use the refinement of sort-merge join discussed in class if possible. Explain your reasoning. [*4 marks*]

(c) Consider the database CQ evaluation

$$\begin{aligned} D \;=\; \{ & R(a,a,c), R(b,a,c), R(c,a,c), \\ & P(a,b,a,c), P(b,a,a,c), P(b,c,a,c), P(c,d,a,c), P(c,e,a,c), \\ & S(a,a,b,c), S(b,b,a,c), S(c,c,b,c), S(d,d,c,c), S(e,e,c,c) \} \end{aligned}$$

and the conjunctive query

$$Q(y,x) :- \; R(x,a,z), P(x,y,a,z), S(y,y,x,z)$$

where x, y, z are variables, and a is a constant value.
Compute the set of tuples $Q(D)$. In addition, for each tuple t of $Q(D)$, provide the match of Q in D that witnesses the membership of t in $Q(D)$. [*5 marks*]