Xi'an Jiaotong-Liverpool University

PAPER CODE	EXAMINER	DEPARTMENT	TEL
CPT201		Computing	

1st **SEMESTER** ***2024/2025 RESIT*** **EXAMINATION**

DATABASE DEVELOPMENT AND DESIGN

TIME ALLOWED: ***2 Hours***

INSTRUCTIONS TO CANDIDATES

1、 **This is a** ***closed-book*** **examination,** ***which is to be written without books or notes*****.**

2、 **Total marks available are 100.**

3、 **This exam consists of** ***two*** **sections:**

Section A consists of 20 short answer questions worth 2 marks each for a total of 40 marks.

Section B consists of 2 problem-solving and quantitative questions worth 30 marks each for a total of 60 marks.

4、 **Answer all questions. There is NO penalty for providing a wrong answer.**

5、 **Only English solutions are accepted. Answer should be written in the answer booklet(s) provided.**

6、 **All materials must be returned to the exam invigilator upon completion of the exam. Failure to do so will be deemed academic misconduct and will be dealt with accordingly.**

Section A: Short Answer Questions

[40 marks]

1) A relation named *product* has 10,000 tuples, which are stored as fixed length and fixed format records; each has the length of 50 bytes. Tuples contain the key attribute *productID* with length of 2 bytes. The tuples are stored sequentially in a number of blocks, ordered by *productID*. Each block has the size of 4,096 bytes and each tuple is fully contained in one block. What would be the number of disk blocks needed to store the relation *product*?

$\lfloor \frac{4096}{50} \rfloor = 81 \quad \lceil \frac{10000}{81} \rceil = 124$

[2/40]

2) With the same information in Part A, Question 1), suppose that a sparse primary index (i.e. one index entry for one block) on *productID* is to be created. A 10-byte pointer to the actual tuples is needed for each index entry. Each index entry is also fully contained in one block. What would be the number of disk blocks needed to store the entire index?

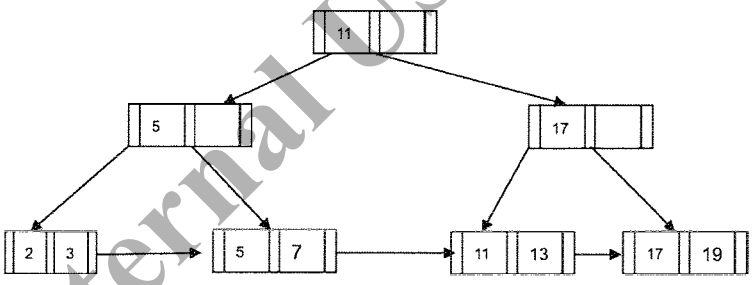
$10+2=12 \quad \lfloor \frac{4096}{12} \rfloor = 341 \quad \lceil \frac{124}{341} \rceil = 1$

[2/40]

3) With the following B+tree, briefly describe the process of looking up the search key "13".

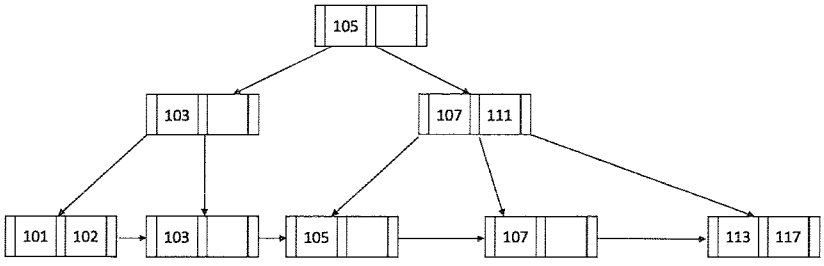
[2/40]

Initial B+ tree



4) Consider the following initial B+ tree index, draw the updated B+ Tree after deleting the search key "107".

[2/40]



5) Briefly state two advantages of extendable hash indexing.

[2/40]

6) Briefly describe how the comparative selection $\delta_{A \leq V}(r)$ can be efficiently evaluated using a secondary index, where A stands for the attribute name of the relation r, and V for a constant.

[2/40]

7) When evaluating selection using a B+ tree index on the candidate key with equality, what is the cost in terms of block transfers and seeks, respectively?

[2/40]

8) Which of the following three algorithms is likely to be more efficient in performing join operations: (1) Indexed nested-loop Join; (2) Merge-Join; (3) Block nested-loop join? Why?

[2/40]

9) What is the main purpose of query optimisation in relational database systems?

To minimize the cost.

[2/40]

10) What are the four properties of transactions in relational database systems?

Atomicity Consistency. Isolation. Durability.

[2/40]

11) Is a conflict serialisable schedule recoverable?

It is not decidable.

[2/40]

12) What is meant by 'a schedule is cascadeless'?

When the Transaction Ti reads the item written by Tj before, Tj must commit before Ti reads.

[2/40]

13) In relational database concurrency control, what needs to be done if deadlocks are detected?

Select a transaction as victim, and rollback it and prevent starvation.

[2/40]

14) In log-based recovery algorithm, what is meant by 'undoing' and 'redoing' a log record of the form '<Ti, X, V1, V2>', where T stands for a transaction, X for the data item, and V for values?

undoing: the X return to its origin value V_1

redoing: the X is updated to V_2 again.

[2/40]

15) What are the two sets of actions that a log-based recovery algorithm needs to take?

Redo

Undo.

[2/40]

16) Briefly explain Asynchronous Replication in distributed database systems?

copies of modified relations are updated over a period of time.

[2/40]

17) Name two techniques for efficiently computing *distributed* joins.

[2/40]

18) Briefly state why the eventual consistency model is preferred in key-value stores for big data.

[2/40]

19) Briefly describe how data is stored in NoSQL databases, e.g. MangoDB?

[2/40]

20) Briefly describe the key advantage of self-supervised learning over supervised learning.

[2/40]

Section B: Problem-Solving and Quantitative Questions

[60 marks]

Question 1. A small social Website has the following three relations in its database.
user (userID, name, email, city, homepage)
liked (userID, PID)
post (PID, title, topic, content, time, date)

"*userId*" is the candidate key for the relation *user*; "*PID*" is the candidate key for the relation *post*; the two attributes in the relation *liked* are the foreign keys referencing *user* and *post*, respectively. In all relations, no records span over one block. The relevant catalog information is given below.

- Number of records in *user*, n_{user} = 300; number of blocks, b_{user} = *50*;
- Number of records in posts, n_{liked} = 8,000; number of blocks, b_{liked} = *70*;
- Number of records in *post*, n_{post} = 1,500; number of blocks, b_{post} = *300*;

Answer the following questions.

[30 marks]

a) Assume that in the worst case, the memory can only hold one block for any of the relations. Using the blocked nested loop join algorithm and the small relation as the outer relation, how many block transfers and seeks would be needed to evaluate "*liked* ⋈ *post*", respectively?

transfer: 300+1500×70 = 105300

seek: 1500+300 = 1800

[6/30]

b) Assume that there is primary B+ tree index with height three (i.e. h_i=3) on *PID* in relation *liked*. Using the indexed nested loop join algorithm, how many block transfers and seeks at minimum would be needed to evaluate "*post* ⋈ *liked*", respectively?

transfer: 300+(1500×(3+1)) = 6300

seek: 300+(1500×(3+1)) = 6300

[6/30]

c) Assume that relation *post* needs to be sorted on key *PID* by using the external merge-sort algorithm. Also, assume memory size, M=10 blocks, and buffer size, b_b=2 blocks. How many block transfers and seeks would be needed, respectively?

Run = $\frac{300}{10}$ = 30

P = $\lceil \log_{10-1}(30) \rceil$ = 3

transfer: 300×2 + 2×1500×3 − 300 = 2100

seek: 2×30 + 2×$\lceil \frac{300}{2} \rceil$×3 − $\lceil \frac{300}{2} \rceil$ = 810

[6/30]

d) Assume that both relations *posts* and *liked* are already sorted on disk. With the merge join algorithm and buffer size b_b=2 blocks, how many block transfers and seeks would be needed to evaluate "*post* ⋈ *liked*", respectively?

transfer: 300+70 = 370

seek: $\lceil \frac{300}{2} \rceil$ + $\lceil \frac{70}{2} \rceil$ = 185

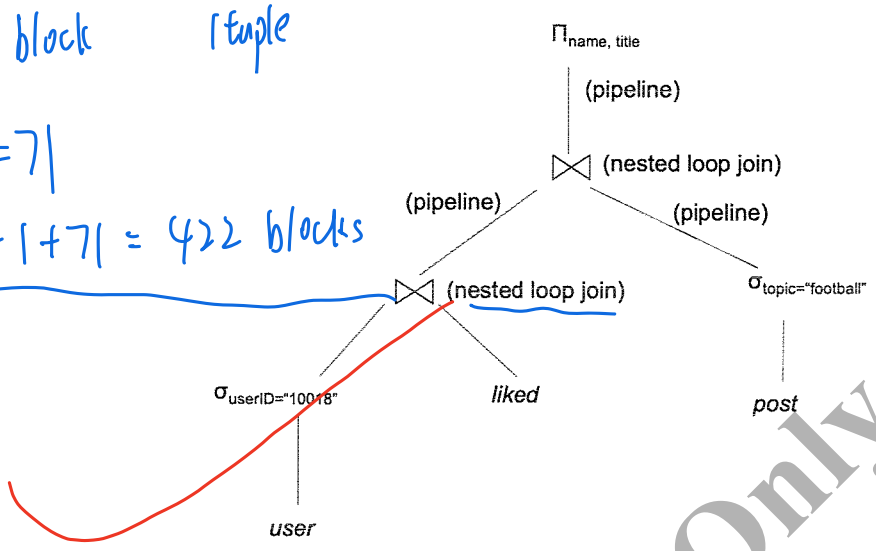
[6/30]

e) An optimised evaluation plan is shown in the following diagram. Assume that linear scan is used to evaluate all the selection operations. The use of pipelining is also shown in the diagram. What would be the total number of block transfers for the whole evaluation plan? Justify your answer.

$\lceil \frac{52}{307} \rceil = 1$ block 1 tuple

$1+1\times 70 = 71$

$50+300+1+71 = 422$ blocks



[6/30]

Question 2. Answer the following questions.

[30 marks]

a) Draw a precedence diagram for the following schedule. Is the schedule conflict serialisable? If yes, to which serial schedule is it equivalent? Justify your answer.
Schedule: T1:read(A); T1:write(A); T1:read(B); T2:read(A); T3:write(B); T2:read(C); T4:read(B); T4:read(D); T4:write(D); T2:read(D).

$T_1 \to T_2$, $T_1 \to T_3$, $T_3 \to T_4$, $T_4 \to T_2$ Yes $T_1 \to T_3 \to T_4 \to T_2$ **[10/30]**

b) Is the schedule in Question 2.a) above view serialisable? If yes, to what serial schedule is it equivalent? Justify your answer.
Yes. $T_1 \to T_3 \to T_4 \to T_2$ **[5/30]**

c) Is the schedule in Question 2.a) cascadeless? Justify your answer.
No. There isn't commit or abort. **[4/30]**

d) Consider the following schedule. Assume the initial values for data items, A, B, and C all equal 100 (i.e. A=B=C=100). Answer the following questions.
(1) What happens with transaction T2? T_2 abort
(2) What log records would be produced by transaction T2? $\langle T_2, B, 100 \rangle$
(3) Which transaction(s) is in the checkpoint L? T_1, T_2, T_3
(4) In the redo pass, which transactions need to be redone? T_3, T_4
(5) After the redo pass, which transaction(s) is left in the undo-list? T_4
(6) In the undo pass, which transactions need to be undone? T_4

(7) After successful recovery, what logs need to be added to the stable storage?

T1	T2	T3	T4
T1 start			
A=A-30			
Write(A)			
	T2 start		
	*B=B*2*		
	Write(B)		
		T3 start	
----------Checkpoint L{}----------			
		C=C+90	
		Write(C)	
		commit	
commit			
			T4 start
			A=A-40
			Write(A)
	B=100		
	abort		
----------System crash----------			

[11/30]

END OF EXAM PAPER