Xi'an Jiaotong-Liverpool University

PAPER CODE	EXAMINER	DEPARTMENT	TEL
CPT201		Computing	

1st SEMESTER *2024/2025 FINAL* 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) Relation *employees* has 5,000 tuples, which are stored as fixed length and fixed format records. Each tuple has the length of 110 bytes and contains the non-key attribute *name* with length of 10 bytes and key attribute *employeeID* with length of 4 bytes. The tuples are stored sequentially in a number of blocks, ordered by *name*. Assume that each block has the size of 8,192 bytes and each tuple is fully contained in one block. What is the number of disk blocks needed to store the relation *employees*?

tuple / blocks = $\lfloor \frac{8192}{110} \rfloor = 74$ total = $\lceil \frac{5000}{74} \rceil = 68$ **[2/40]**

2) With the same information in Part A, Question 1), suppose that a dense secondary index on the *employeeID* attribute is to be created. A 10-byte pointer to the actual tuple is needed for each index entry. Each index entry is also fully contained in one block. What would be the number of blocks needed to store the index?

4+10=14 bytes $\lfloor \frac{8192}{14} \rfloor = 585$ total = $\lceil \frac{5000}{585} \rceil = 9$ **[2/40]**

3) With the same information in Part A, Question 2), would a sparse secondary index on *employeeID* be more efficient than a dense primary index on the non-key attribute *name*? Why?

No.

[2/40]

4) Briefly state the main disadvantage of static hashing for indexing in database systems.

fixed size **[2/40]**

5) As an important data structure for spatial index, an R-tree is useful for indexing sets of rectangles and polygons. Briefly describe a potential real-world application of R-tree.

Information System. Geographic **[2/40]**

6) In the context of query optimisation, what is meant by materialised evaluation?

Intermediate results are stored. **[2/40]**

7) Suppose a relation r contains an attribute A, and the number of distinct values that appear in r for attribute A is V(A,r). What would be the size of the projection $\Pi_A(r)$?

V(A,r) **[2/40]**

8) Suppose R and S are the attributes of relations r and s, respectively. Assume that $R \cap S$ is a foreign key in S referencing R, how to estimate the size of the join "r $\bowtie$ s"?

The number of tuples in S **[2/40]**

9) One of the rules used in heuristic optimisation is "perform selection early". Briefly explain why it can help improve execution performance.

There will be less tuples. So the operations next will be less. **[2/40]**

10) In the context of transaction management in relational database systems, briefly describe what is meant by *atomicity*.

Either all of the operations are properly reflected in database or none is. [2/40]

11) Transactions can be in different states, i.e. *Active*, *Partially committed*, *Failed*, *Aborted* and *Committed*. Briefly describe what is meant by the state *Aborted*.

The transaction has been rolled back and the data are stored as the prior to start of transaction. [2/40]

12) Briefly describe how to test if a schedule is conflict serialisable.

Use procedence graph. [2/40]

13) Briefly describe the two-phase locking protocol for concurrency control in relational databases.

Growing phase
Shinking phase. [2/40]

14) In the context of relational databases, what is meant by a *deadlock*?

two transaction asked for each other. [2/40]

15) In relational databases implementing a log, what log records need to be written to the stable storage if there is a transaction rollback during normal transaction processing?

short [2/40]

16) Briefly describe the main difference between the Two-Phase Locking Protocol and Two-Phase Commit Protocol.

[2/40]

17) What are the four general principles in linked data design and publication?

[2/40]

18) What is the main difference between 'Strong consistency' and 'Eventual Consistency' in database systems?

[2/40]

19) Name two popular categories of NoSQL databases.

[2/40]

20) Briefly describe what self-supervised learning is in the context of natural language processing.

[2/40]

Section B: Problem-Solving and Quantitative Questions

[60 marks]

Question 1. Consider the following three relations in a library system and their catalog information.

student(SID, name, email, programme, year, department)
borrows(SID, BID)
book(BID, title, authors, ISBN, publisher, category)

- *SID* is the key for *student*, and *BID* is the key for *book*;
- *borrows.SID* and *borrows.BID* are the foreign keys referencing *student* and *book*, respectively;
- number of records in *student*, $n_{student}$ = 800; number of blocks in *student*, $b_{student}$ =100;
- number of distinct values for the attribute *department* in the *student* relation, *V(department, student) = 50;*
- index: a four-level primary B$^+$-tree index (height=4) on the *SID* attribute of *borrows* relation.
- number of records in *borrows*, $n_{borrows}$ = 10,000; number of blocks in *borrows*, $b_{borrows}$ = 50;
- number of records in *book*, n_{book} = 3,000; the number of blocks in *book*, b_{book} = 600;

Answer the questions below.

[30 marks]

a) Assume the worst case, using the nested loop join algorithm and relation *student* as the outer relation to evaluate "*student* ⋈ *borrows*", how many block transfers and seeks would be needed, respectively? Justify your answer.

transfer: 100 + 800 × 50 = 40000 + 100 = 40100
seek: 100 + 800 = 900

[4/30]

b) Using the block nested loop join algorithm to evaluate "*student* ⋈ *borrows*", which relation is preferred to be the outer relation? How many block transfers and seeks would be needed, respectively? Justify your answer.

borrows transfer: 50 + 50 × 100 = 5050
seek: 50 × 2 = 100

[6/30]

c) Using the indexed nested loop join algorithm (with the available B+ tree index on *borrows*) to evaluate "*student* ⋈ *borrows*". How many block transfers and seeks would be needed, respectively? Justify your answer.

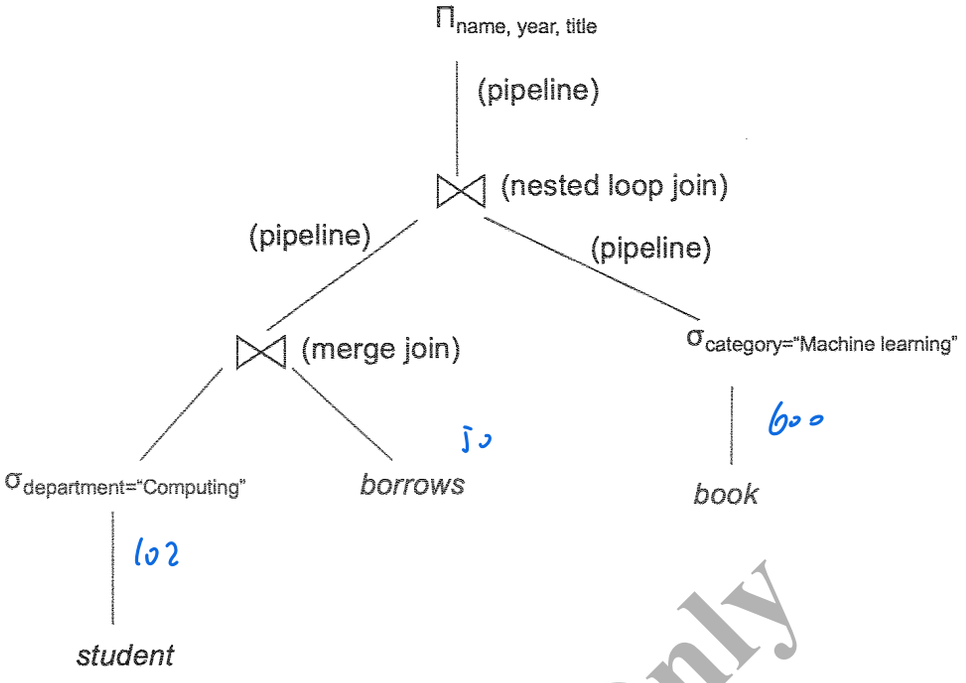
transfer: 100 + 800 (4+1) = 4100
seek: 100 + 800(4+1) = 4100

[4/30]

d) Suppose an optimised query plan has been formulated, as shown in the diagram below. Write the corresponding relational algebra expression.

$\Pi_{name,email}$ (($\sigma_{department="computing"}$ student) ⋈ borrows) ⋈ ($\sigma_{category="ML"}$ book))

[4/30]



e) With the optimised plan in the diagram in Question 1.d), assume that linear scan is used to evaluate all the selection operations, and relation *student* has already been sorted on the key *SID*. No intermediate relations need to be stored as the result of using pipelining. Estimate the total evaluation cost in terms of the number of block transfers. Justify your answer.

[6/30]

f) If the merge join is replaced by indexed nested loop join in the query plan in Question 1.e), and other information remains unchanged, would this new plan be more efficient? Justify your answer.

[6/30]

Question 2. Answer the following questions.

[30 marks]

a) Consider the schedule below. Draw a precedence diagram for the schedule. Is it conflict serialisable? Justify your answer.

[8/30]

T1	T2	T3	T4
	read(X);		
	read(Y);		
			read(X);
			read(Y);
		read(X);	
			write(X);
	write(Y);		
	read(Z);		
			write(Y);

		write(X);	
read(Y);			
write(Y);			

b) Is the schedule in Question 2.a) recoverable? Justify your answer.

No. there is no commit.

[4/30]

c) Is it possible that a cascadeless schedule contains a blind write? Justify your answer (you may provide an example).

Yes.

T_1 T_2
R(x)
W(x)
commit
W(x)
commit

[6/30]

d) Consider the following partial schedule with log-based recovery. Assume that the initial values for X and Y are both 10 (i.e. X=Y=10). Answer the following questions:

(1) What are the transactions in the checkpoint L1{}? T_1, T_3

(2) A crash happens immediately at time=15. When recovering from the system crash, in the redo pass, which transactions need to be redone? T_1 T_2 T_3

(3) After the redo pass, what transactions are left in the undo list? T_1 T_2

(4) Which transactions need to be undone during the recovery? T_1 T_2

(5) What logs need to be inserted to stable storage after the successful recovery?

$\langle T_2, Y, 3 \rangle$ to, $\langle T_1, X, 10 \rangle$, $\langle T_2 \text{ abort} \rangle$, $\langle T_1, \text{abort} \rangle$

Time	T1	T2	T3
0			start
1	start		
2	read(X)		
3	----------**Checkpoint L1{…}**----------		
4			read(Y)
5			Y=Y/3
6			write(Y)
7		start	
8		read(Y)	
9		read(X)	
10		Y=Y+X	
			commit
11	X=X*2		
	write(X)		
12		write(Y)	
13		X=X-10	
14		read(Y)	
15	----------**Database Failure**----------		

[12/30]

END OF EXAM PAPER