Xi'an Jiaotong-Liverpool University

PAPER CODE	EXAMINER	DEPARTMENT	TEL
CPT201		Computing	

FIRST SEMESTER 2022/2023 RESIT EXAMINATIONS

BACHELOR DEGREE – Year 3

DATABASE DEVELOPMENT AND DESIGN

Exam Duration: ***2 Hours***

Crash Time Allowed: ***30 Minutes (for online exam only)***

INSTRUCTIONS TO CANDIDATES

1、 **This is a close-book exam. Please tick the integrity disclaimer *when uploading your answers on LM Core* and complete the assessment independently and honestly.**

2、 **Total marks available are 100. This will count for 70% in the final assessment.**

3、 **Answer ALL questions in both Part A and Part B.**

4、 **The number in the column on the right indicates the marks for each section.**

5、 **The university approved calculator - Casio FS82ES/83ES can be used.**

6、 **All the answers must be in English.**

7、 **The duration is *2 hours*, and an additional *30-minute* crash time beyond the exam duration will be allowed for you to report and resolve minor technical issues which may be encountered during the online exam. Where there are any major problems preventing you from continuing the exam or submitting your answers in time, please do not hesitate to email the Assessment Team of Registry (assessment@xjtlu.edu.cn).**

THIS PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

PART A: Short Answer Questions

[40 marks]

1) Suppose that there are 30,500 tuples in a relation called *employee*. Tuples are stored as fixed length and fixed format records; each of the length of 120 bytes. Tuples contain the non-key attribute *name* with the length of 20 bytes. All tuples are stored sequentially in blocks, ordered by *name*. Each block is of the size of 4,096 bytes and no tuple spans over one block. Compute the total number of blocks needed to store the relation *employee.*

$\lfloor \frac{4096}{120} \rfloor = 34$ $\lceil \frac{30500}{34} \rceil = 898$

[2/40]

2) With the same information in Part A, Question 1), suppose that a primary B+tree index on the *name* attribute is to be created. A 15-byte pointer to actual tuples (a 10 byte block id and 5 byte offset) is needed for each index entry. No index entry spans over one block. Assume that only one index entry is needed for one block. Compute the number of blocks needed to store the index.

$15+20=35$ bytes $\lfloor \frac{4096}{35} \rfloor = 117$ $\lceil \frac{898}{117} \rceil = 8$

[2/40]

3) In B+ tree deletion algorithm, when should "merge siblings" and "redistribute pointers" be used, respectively?

[2/40]

4) Briefly describe two types of queries that can be effectively and efficiently supported by spatial indexing techniques.

range query

nearest-neighbour query

[2/40]

5) In evaluating selection operations using a primary index (e.g. B+ tree) on the candidate key with equality, what is the cost in terms of block transfers and seeks, respectively?

transfer: hi+1 seek: hi+1

[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.

scan leaf pages to get the pointers to records until the index entry > V.

[2/40]

7) Which of the following three join algorithms is the most efficient in terms of the number of block transfers and seeks: (1) Block-Nested-Loop Join; (2) Indexed-Nested-Loop Join; and (3) Merge-Join? Why?

[2/40]

8) What is the major difference between 'materialisation' and 'pipelining' in query optimisation in relational database systems?

[2/40]

9) In estimating the size of join, let r and s denote two relations, and R and S are the attributes of r and s, respectively. If $R \cap S$ is a foreign key in S referencing R, what would be the number of tuples in $r \bowtie s$?

[2/40]

10) What is meant by atomicity and durability of transactions in relational database systems?

[2/40]

11) Is a view serialisable schedule always conflict serialisable? Why?

[2/40]

12) Briefly describe how to determine if a schedule is recoverable.

[2/40]

13) Starvation is possible under the two-phase locking protocol if the concurrency control manager is badly designed. Briefly describe a solution to solve this problem.

[2/40]

14) What is meant by 'redoing' and 'undoing' transactions in log-based failure recovery algorithms with checkpoints, respectively?

[2/40]

15) Briefly describe the two types of replication strategies in distributed database systems.

[2/40]

16) Briefly describe how deadlocks can be detected with a centralised approach in distributed database systems.

[2/40]

17) Briefly describe the data model for an RDF (Resource Description Framework) triple.

[2/40]

18) What is meant by 'Eventual Consistency' in many of the contemporary big data storage systems?

[2/40]

19) There are a number of properties for blockchain based storage systems. Two of them are called 'Tamper resistance' and 'irrefutability'. Briefly describe what is meant by these two terms, respectively.

[2/40]

20) Association rule analysis is a common technique for data analytics in database systems. Briefly describe the difference of 'Support' and 'Confidence' in association rule analysis.

[2/40]

PART B: Problem-Solving and Quantitative Questions

[60 marks]

Question 1. A digital library maintains the following relations in its database.

- *author* (*<u>authorID</u>, name, affiliation, email); candidate key 'authorID';*
- *paper (paperID, title, conference, year); candidate key 'paperID'*
- *publishes* (*<u>paperID</u>, authorID); foreign keys 'authorID' and 'paperID'* referencing relations *author* and *paper;*

In relation *author*, there are 6,000 tuples stored on 300 blocks; in relation *paper*, there are 90,000 tuples stored on 7,000 blocks; and in relation *publishes*, there are 130,000 tuples stored on 200 blocks. A primary B^{+}-tree index of height 4 on the *authorID* attribute of the *publishes* relation has been created.

Answer the following questions.

[30 marks]

a) Describe how the selection $\delta_{year>2022}$ on relation *paper* can be evaluated in the most cost effective way by using a primary B+ tree index on *year* and a secondary B+ tree index on *year*, respectively.

[4/30]

b) Assume that the memory can only hold one block for any of the relations. Using the blocked nested loop join algorithm and the smaller relation as the outer relation, how many block transfers and seeks would be needed to evaluate "*author* ⋈ *publishes*", respectively?

transfer: 200 + 200×300 = 60200

seek: 200×2 = 400

[4/30]

c) Assume that the indexed nested loop join algorithm is used based on the available B+ tree index on *publishes*. How many block transfers and seeks at minimum would be needed to evaluate "*author* ⋈ *publishes*", respectively?

transfer: 300 + 6000×5 = 30300

Seek: 300 + 6000×5 = 30300

[4/30]

d) Describe how "*author* ⋈ *publishes*" can be evaluated with the hash join algorithm.

[4/30]

e) Discuss which algorithm (i.e. block nested look join, indexed nested look join and hash join) would be the most efficient in evaluating the join operation based on results from Question 1. b), c) and d).

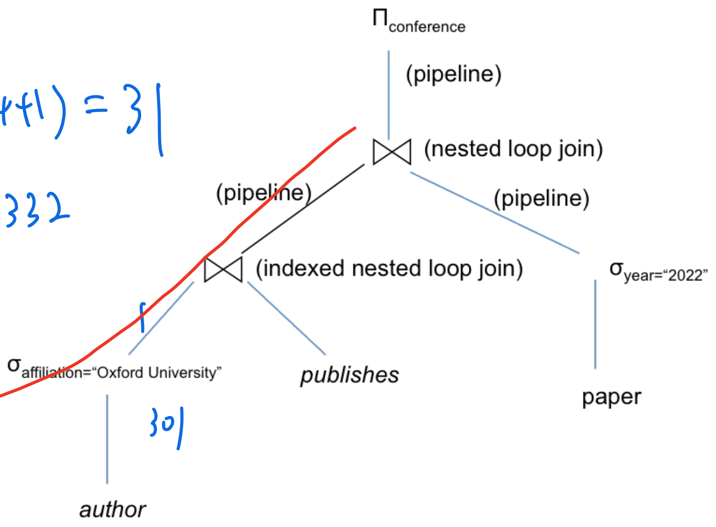
[4/30]

f) An optimised query evaluation plan is shown in the diagram below. It is known that number of distinct values for the attribute *affiliation* in the *author* relation, *V(author, affiliation)* = 1,000. Assume that linear scan is used to evaluate all the selection operations. The use of pipelining is also shown in the diagram. What is the total number of block transfers for the whole evaluation plan? Justify your answer.

300 + 1 = 301

1 + (6000 ÷ 1000) × (4+1) = 31

301 + 31 + 7000 = 7332



[10/30]

Question 2. Answer the following questions.

[30 marks]

a) Draw a precedence diagram for the partial schedule below. Is the schedule conflict serialisable? Justify your answer.
Schedule: T1:write(A); T1:read(B); T1:read(C); T3:write(B); T1:write(B); T3:write(C); T2:read(C); T2:write(B); T2:write(C); T3: read(A)

$T_1 \rightleftarrows T_3$, $T_1 \rightarrow T_2$, $T_3 \rightarrow T_2$

No.

There is a cycle.

[6/30]

b) Is the following partial schedule view serialisable? If yes, to what serial schedule is it equivalent? Justify your answer.
Schedule: T1:read(A); T2:write(A); T1:write(A); T3:write(A);

Yes. T_1 T_2 T_3

[6/30]

c) Is the following partial schedule recoverable? Justify your answer.
Schedule: T1:write(X); T1:write(Y); T2:read(X); T2:write(Y); T2:abort; T1:write(Z); T1:commit; T3:read(Y); T3:write(Z); T3:commit.

No. T_1 commit after T_2.

[4/30]

d) What is meant by ‘a schedule is cascadeless’?

T_i read T_j's written data, T_j commit before T_i read.

[4/30]

e) Consider the following partial schedule. If the database failure happens at time=18. Answer the following questions: (1) Which transaction(s) is in the checkpoint L1? (2) In the redo pass, which transactions need to be redone? (3) After the redo pass, which

(1). T_3 (2). T_1 T_2 T_3 (3). T_1 (4). T_1 (5). <T_1 X X> <T_1 abort>

transaction(s) is left in the undo-list? (4) In the undo pass, which transactions need to be undone? (5) After the successful recovery, what logs need to be added to the stable storage?

Time	**T1**	**T2**	**T3**
0			start
1			read(Y)
2	**---------------------------*Checkpoint L*--------------------------**		
3			Y=Y+1
4	start		
5	read(X)		
6	X=X+1		
7			write(Y)
8			commit
9		start	
10		read(X)	
11		read(Y)	
12		Y=Y+X	
13		write(Y)	
14		commit	
15	read(Y)		
16	Y=Y+X		
17	write(X)		
18	**-------------------------System Failure-------------------------**		

[10/25]

END OF EXAM PAPER