

DI- FCT/UNL

December 15<sup>th</sup>, 2025

# Database Systems

**2nd test 2025/26 – Version C**

**Duration: 2 hours (limited consultation)**

**Group 1 (each question is worth 2,5 values out of 20)**

Consider part of a database for a supermarket, with a single store, consisting of six tables:

Employee( <u>emp_id</u> , name, position)	Supplier( <u>supplier_id</u> , supplier_name, contact_phone)
---	--

Product( <u>product_id</u> , product_name, price, supplier_id)	Customer( <u>customer_id</u> , name, phone)
--	---

Sale( <u>sale_id</u> , sale_date, customer_id, emp_id).	Sale_Item( <u>sale_id</u> , <u>product_id</u> , quantity)
---	---

These tables capture staff, suppliers, products, customers, completed sales, and detailed sale line-items for each transaction in the supermarket. The primary keys are emp\_id for Employee, supplier\_id for Supplier, product\_id for Product, customer\_id for Customer, and sale\_id for Sale. Sale\_Item uses a composite primary key (sale\_id, product\_id). Regarding Foreign keys, Product.supplier\_id references Supplier, Sale references Customer and Employee, and Sale\_Item references both Sale and Product. This ensures that every product has a valid supplier, every sale has a valid customer and employee, and every sale item belongs to an existing sale and product.

The adopted DBMS uses blocks of 8KiB (8192 bytes). The records of Sale and Sale\_Item have fixed length, while the other tables have variable length size. Estimated cardinalities, sizes, and 8-KiB block counts are: Employee with about 100 rows (90 bytes per tuple/2 blocks), Supplier with about 50 rows (72 bytes/1 block), Product with about 5,000 rows (70 bytes/43 blocks), Customer with about 20,000 rows (72 bytes/176 blocks), Sale with about 500,000 rows (20 bytes/1,221 blocks), and Sale\_Item with about 2,000,000 rows (16 bytes/3,907 blocks). For each of these tables there is a secondary B+ tree (non-clustering) index on the primary key attribute(s), created with the column order indicated in the tables.

A B+ tree node can contain about 500 search keys, and it is known that seek time  $t_s$  is 9ms and the transfer time of a block  $t_T$  is 1ms, while the memory only holds 100 blocks.

**Note:** In this group, whenever examples are requested, these must be exclusively about this database,  
Additionally, all the answers must contain a brief justification.

- 1 a)** Determine the estimated number of tuples returned for the following query, knowing that the primary key of employee is numbered sequentially and starting from 1

```

SELECT *
FROM (SELECT * FROM Employee WHERE emp_id IN (1,2,3) )
INNER JOIN Sale USING (emp_id)
INNER JOIN Customer USING (customer_id)

```

- 1 b)** Consider the following schedule of two SQL transactions. Place the lock-S, lock-X, and unlock instructions in this schedule knowing that the schedule obeys to rigorous two-phase locking protocol. Indicate the steps, if any, where the transactions are blocked or unblocked.

	<b>Transaction T1:</b>	<b>Transaction T2:</b>
<b>1</b>		UPDATE Product SET price=47.50 WHERE product_id = 486
<b>2</b>	UPDATE Product SET price=54.60 WHERE product_id = 176	
<b>3</b>		SELECT AVG(price) FROM products WHERE product_id BETWEEN 400 AND 499
<b>4</b>	ABORT	
<b>5</b>		UPDATE Product SET price=54.65 WHERE product_id = 176
<b>6</b>		COMMIT

- 1 c)** Consider the following recovery log record of a DBMS with memory buffering of disk blocks in which disk writes are only performed when checkpoints are performed (you can assume that there is no log buffering). The memory is initially empty. Complete the log knowing that the system has recovered from the recorded failure, indicating for each step the status of the disk and memory, as well as the values of L1 and L2.

	Log
1	<T1, start>
2	<T2, start>
3	<T3, start>
4	<T1,Product(127).price,10.50,11.00>
5	<T1,Product(128).price,10.50,12.00>
6	<T2,Product(129).price,10.70,10.80>
7	<T2, commit>
8	<checkpoint L1>
9	<T3,Product(129).price,10.80,10.70>
10	<T1,Product(130).price,20.50,20.60>
11	<checkpoint L2>
12	<T3, commit>
13	CRASH

**1 d)** Transactions in the schedule below are to be executed in SNAPSHOT ISOLATION mode with first-committer wins policy. At start, table Employees contains only the tuple (1,'Alice','Administrator'), and table N is empty. Display the contents of table N at the end of the run knowing that it only has one column, and no integrity constraints are in effect. Verify whether the resulting schedule of the SUCCESSFULLY executed transactions is serializable. Note that transactions may be rolled back and that the answer must reflect the effects of successful and aborted transactions.

Step	Transaction T1:	Transaction T2:	Transaction T3:
1		begin transaction UPDATE Employee SET name='Alan' WHERE emp_id = 1	
2		INSERT INTO N SELECT COUNT(*) FROM Employee;	
3	begin transaction INSERT INTO Employee VALUES(2,'Bob','Clerk')		
4	INSERT INTO N SELECT COUNT(*) FROM Employee;		
5			begin transaction INSERT INTO Employee VALUES(3,'Charles','Clerk')
6	UPDATE Employee SET name='Helen' WHERE emp_id = 1		
7	COMMIT		
8			INSERT INTO N SELECT COUNT(*) FROM Employee;
9			COMMIT
10		COMMIT	

**1 e)** Consider the following distributed transaction that takes place at three different locations. Indicate the concurrency control log records (in order) of the coordinator and of the transaction managers of the several sites knowing that the transaction is coordinated by site 3, that the transaction coordinator failed after receiving from site 1 the decision to abort (but the coordinator did not log the abort message), and that it is used 2 two-phase commit protocol for concurrency control. Explain what is the decision of site 2 when it discovers that the coordinator has failed before it recovers.

#	Site 1	Site 2	Site 3
1			begin transaction
2	UPDATE Product SET price=54.60 WHERE product_id = 176		
3			UPDATE Employee SET ... WHERE emp_id = 765
4		SELECT COUNT(*) FROM Sales	
5			COMMIT

**Group2 (each question is worth 2,5 values out of 20)**

**Note:** the answer to each question cannot exceed 1 page

- 2 a)** Suppose we intend to perform the natural join of two tables r and s with a common attribute A, which is neither a key for R nor for S. In this case, the estimated number of tuples for the join is given by the expression:

$$\min \left( n_r * \frac{n_s}{V(A, s)}, n_s * \frac{n_r}{V(A, r)} \right) = \min \left( \frac{n_r * n_s}{V(A, s)}, \frac{n_r * n_s}{V(A, r)} \right)$$

Explain the rationale of the formula (you can provide examples).

**HINT:** You can imagine a join as tuples in one table selecting tuples in the other table.

- 2 b)** Describe the strict two-phase locking protocol and show that the resulting schedules allowed by the protocol do not suffer from cascading rollbacks.
- 2 c)** The recovery algorithm supports the no-force policy and the steal policy. Define what are these policies, and corresponding advantages or disadvantages.

# THE END