HOCHSCHULE FÜR TECHNIK STUTTGART
UNIVERSITY OF APPLIED SCIENCES

MASTER'S COURSE **S**oftware **T**echnology
STUDIENGANG **M**aster **M**athematik

EXAMINATION in **SUMMER SEMESTER 2009**

MODULE:	**Databases / Databases 2**	NAME:	
DATE:	**6 July 2009**	SEMESTER:	
TIME:	**11.00 - 13.00 Uhr**		
EXAMINER:	**Prof. D. Koch**		

ALLOWED AIDS:	All materials that were distributed in class and in the Blackboard, all your own notes, two text books of your choice, plus an English dictionary.
NOT ALLOWED:	Mobile Phones, laptop, and other communication devices
APPENDIX:	Stored Procedure Code

Please write your name on each sheet that you turn in.

Turn in the problem sheets as well!

Problem	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	Σ
Maximum points	8	10	10	6	12	8	10	18	10	12	106
Achieved points											
Grade											

Problem 1. (8 points)

Given the problem of retrieving a single data record by its primary key, what is the method of physical data organization that provides the fastest access for this operation? Briefly explain why.

Problem 2. (2 times 5 points = 10 points)

Imagine you have been hired by a company that works with data that has been managed in an unsatisfactory way so far. They decide to buy a new DBMS and completely reorganize their data. Your task is to make the decision which DBMS to chose (type, vendor).
Name and briefly explain two criteria on which you will base your decision.

Problem 3. (10 points)

Design a relation for the following application:

A company stores in a central database data records that are sent on a case by case basis from different partner companies.

Each transmitted data record is sent as a tuple of String values, along with the respective attribute names.
It is never known beforehand which attributes the next transmitted record will contain. Also, the set of possible attributes in a record is not predefined and not restricted to a certain number of attributes.

Example:
The following data records may be transmitted:

- customerNr: 1345, familyName: Koch, order: laundry machine, address: München
- customerNr: 4711, familyName: Schmidt, order: television
- familyName: Schmidt, customerNr: 1212, order: table, creditworthiness: good, address: Hamburg
- customerNr: 4321, address: Stuttgart, newsletter: yes

Remember: Data records that will be transmitted in the future may contain additional attributes that are not currently known yet.

Define the schema of <u>one</u> relational table in which such data records can be stored. The relation must allow that

- Any transmitted tuple (with attribute names and attribute values) can be stored, no matter how many additional, previously unknown pairs of "attribute name : value" it contains.
- It is recognizable which data records where transmitted and which attribute name / value pairs belong to each record.

Problem 4. (6 points)

Suppose a DBMS does not allow the definition of integrity constraints. What is the consequence for the development of applications accessing this database?

Problem 5. (2 times 6 points = 12 points)

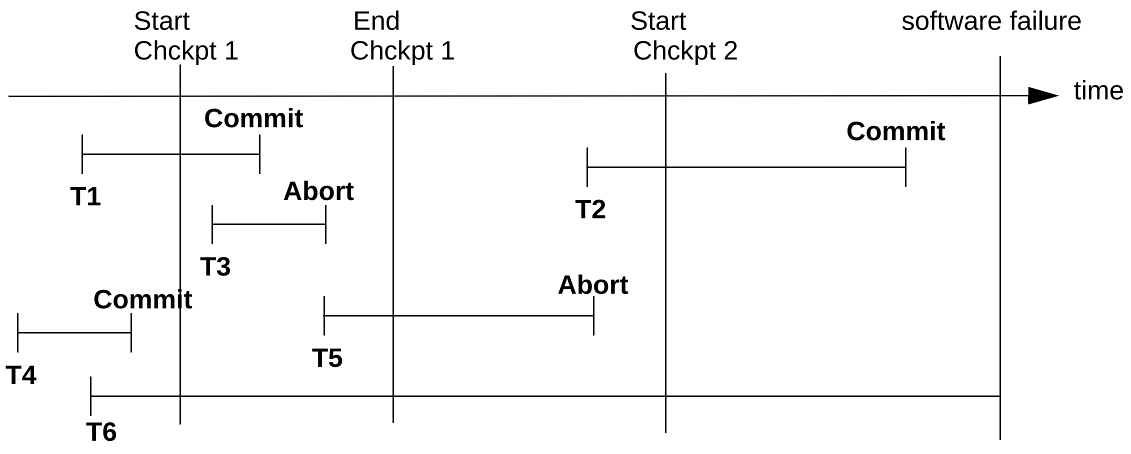
Consider a situation where you need to implement a recursive query in a relational system and the options to do it either with a WITH RECURSIVE clause in SQL or with a recursive stored procedure.
a) Briefly explain one advantage and one disadvantage of the SQL solution.
b) Briefly explain one advantage and one disadvantage of the stored procedure solution.

Problem 6. (8 points)

What is the purpose of the Java Interface ResultSetMetadata?
Give an example of a situation in which it is needed. (You don't need to write code, rather, explain what happens).

Problem 7. (2 times 5 points = 10 points)

Consider the following scenario with a log file where fuzzy checkpointing is used:



a) List which transactions the recovery manager must undo and which ones it must redo. Briefly explain your choice.

b) Now assume that Checkpoint 2 had successfully finished just before the software failure occurred, after the Commit of T2. What is the answer of problem a) now? Briefly explain.

Problem 8. (3 times 6 points = 18 points)

Consider the following example of a dirty read that was discussed in class:

Start transaction A	
A reads a copy of the account balance: copy1 = 1000 EUR	
A withdraws 200 EUR copy1 := 1000 - 200 = 800 EUR	
A writes 800 EUR to the database as the new balance	**Start transaction B**
	B reads a copy of the account balance: copy2 = 800 EUR
	B withdraws 400 EUR copy2 := 800 - 400 = 200 EUR
A aborts and rolls back.	**B writes 200 EUR to the database as the new balance**
	B commits.

a) (6 points)
Write a transaction schedule with read and write operations that demonstrates that a dirty read will not be prevented by locking if the 2PL protocol is not observed.
Hints:

- For simplicity, you don’t need to write down the lock requests, but can assume a lock is immediately granted.
- Please note that the rollback contains a write operation, too!

b) (6 points)
Write a schedule with read and write operations that obeys the 2PL protocol that will prevent the dirty read problem.

c) (6 points)
Briefly explain why serializable schedules are often preferable to serial schedules.

Problem 9. (12 points)

Which ACID properties of transactions are affected by recovery? Explain.

Problem 10. (12 points)

In class we discussed the code of the recursive stored procedure for finding all directr and indirect supervisees of a manager. You find this code in the annex appended here.

a) (6 points)
Briefly explain what the cursor inside this procedure is used for.

b) (6 points)
Do all recursive calls of the routine "rec_iteration" access the identical cursor structure cur1, or do they all maintain a different cursor cur1? Points are only given for a (brief) explanation.

APPENDIX

Code of a recursive stored procedure discussed in class:

```
DROP PROCEDURE IF EXISTS rec_iteration//
CREATE PROCEDURE rec_iteration( IN id INT, IN tablename VARCHAR(128) )
BEGIN

   DECLARE done INT DEFAULT 0;

   DECLARE a INT;

   DECLARE cur1 CURSOR FOR SELECT empNr from Employee where mgrNr  = id;

   DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

   OPEN cur1;

   repeat
      FETCH cur1 INTO a;
      /* now a contains a mgrNr */
      IF NOT done THEN
         SELECT CONCAT("INSERT INTO ", tablename, " VALUES (?)")
                        INTO @stmt_body;
         PREPARE stmt FROM @stmt_body;
         SET @internal = a;
         EXECUTE stmt USING @internal;
         DEALLOCATE PREPARE stmt;
         call rec_iteration(a, tablename);
      end IF;
   UNTIL done END REPEAT;
   CLOSE cur1;
END//
```