Written Exam
Advanced Database Systems
course code: 192110902

6 November 2013 (08:45 - 12:15), CR-2K

Djoerd Hiemstra, Maurice van Keulen, Robin Aly

Remarks:

- Motivate yours answers. The motivation / argumentation plays an important role in grading the assignments.
- You may not consult books or notes, but only one page of A4 size, double-sided. The page may contain text (typed or hand-written) and (possibly reduced) images (copied from the book, other sources or hand-made).
- For each assignment, the number of points is given. They add up to 90. You get 10 points for showing up at the exam. The grade for the exam is determined by dividing the number of points by 10.
- The final grade for the course is determined by $0.7 \times$ 'Grade written exam' $+ 0.3 \times$ 'Grade project'. Both grades need to be ≥ 5.5 and the MiniLab assignment has to be completed satisfactorily.

Question 1: Using SQL in an Application (10 points)

Below you find a fragment of example code.

```
 1 import java.sql.*
 2
 3 # SQL iterator
 4 # GetEnrolledIter(int studentID, String studGrade);
 5 GetEnrolledIter iter1;
 6
 7 # SQL iter1 = { SELECT T.StudId AS studentId,
 8                        T.Grade AS studentGrade
 9                 FROM   Transcript T
10                 WHERE  T.CrsCode = :crsCode
11                 AND    T.Semester = :semester };
12
13 int id;
14 String grade;
15 while (iter1.next()) {
16    id = iter1.studentId();
17    grade = iter1.studGrade();
   ⋮
28 }
   ⋮
```

(a) Is the programme code using a so-called "Statement-Level Interface" or a "Call-Level Interface" to include the SQL constructs? Explain your answer.

(b) Is this form of embedding SQL statements referred to as (answer with "yes" or "no"):

i) dynamic SQL
ii) static SQL
iii) embedded SQL
iv) SQL/PSM
v) a stored procedure

(c) Which line numbers contain a so-called "cursor mechanism"? In this case, the cursor is insensitive. What is an "insensitive cursor"? ← no changes visible

(d) The example code uses host language variables as parameters of SQL queries instead of so-called placeholders. Explain the difference between using host language variables and placeholders. What are the advantages of using host language variables?

Question 2: Physical database design and query optimization (30 points)

The database of a big company (about 10,000 employees, 2000 projects and 5000 project assignments) has the following schema.

```
Employees(ENR, EName, Salary)
Projects(PNR, Pname, PType, Budget)
Assignments(ENR, PNR, Description, Hours, Date)
```

Keys are underlined. Of course, ENR and PNR are foreign keys in Assignments. Every week the following operations are done on the database:

i) "Select all employee, project and assignment information from the database related to projects of a given type, for instance projects of type 'IT' ". The company distinguishes 10 different project types. This operation is done in 5% of the times.

ii) "Select all employee, project and assignment information from the database related to projects with a budget bigger than 10,000 Euro". This query makes 1% of the total number of queries.

iii) "Insert an new assignment": 50% of the queries and "Update or remove an assignment": 37% of the queries.

The company has an RDBMS that supports sorted files, hash indexes (clustered or unclustered), and B$^+$-tree indexes (clustered or unclustered).

Answer the following questions.

(a) Explain why a file can have only one clustered index.

(b) Based on the case above, give an adequate physical database design. Explain your answer.

(c) Formulate operation i) above in SQL, translate the query to relational algebra and perform stepwise heuristic optimization of the algebraic expression based on algebraic equivalences.

(d) Assume that the page size of the RDBMS is 4 kb; that a employee tuple takes about 40 bytes, a project tuple about 80 bytes and an assignment tuple about 1024 bytes.

Based on your physical database design given in Question (b) and your answer to Question (c) above, give an efficient query execution plan and estimate the costs of this plan measured by the expected number of page transfers needed.

Question 3: Recovery (15 points)

Suppose the database crashed and upon restart we find the following situation. The database uses a *no-force commit policy*, pessimistic concurrency control, and sharp checkpointing. The database schema contains an integrity constraint: $C: x \neq y$.

	6	7	8	9	10	11	12	13	14	15	16	17
	T_1	T_1	T_2	T_2	$T_1; T_2$	T_2	T_2	T_1	T_3	T_1	T_1	T_3
Log: ···	B	U	B	U	CK	U	A	U	B	U	C	U
		p		q		q		x		y		x
		0 1		0 1		1 0		5 10		10 5		10 11

Each record from top to bottom:

LSN, transaction id, record type, database variable, before and after image

Type: B:begin transaction, U:update, C:commit, A:abort, CK:checkpoint.

Database pages:	Page 3	Page 8
	LSN:14	LSN:10
	x=10	p=1
	y=10	q=1

(a) The 'C' and 'D' in 'ACID' guarantee consistency and durability. Database page 3 on disk, however, contains an inconsistent state, because it violates integrity constraint C. Does the above information contain an error, i.e., is it impossible to find the above situation on disk after a crash? If so, what is wrong in your opinion? If not, how are consistency and durability guaranteed? Explain your answer.

(b) The recovery protocol reconstructs a consistent database state. Present the steps of the protocol for obtaining a consistent database state in this situation. Also give the reconstructed database state (i.e., values of the variables x, y, p, and q in pages 3 and 8). Explain your answer.

Question 4: Serializability and Isolation Levels (15 points)

Suppose we have the two transactions below. T_1 swaps the values of the two database variables db_x and db_y, and T_2 increases db_x by one, and decreases db_y by one. v, v_1 and v_2 are local variables in the application program (i.e., they are transient and not stored in the database).

$$T_1 \;:\; \text{BEGIN}; \underbrace{v_1 := db_x}_{r_1(x)}; \underbrace{v_2 := db_y}_{r_1(y)}; \underbrace{db_x := v_2}_{w_1(x)}; \underbrace{db_y := v_1}_{w_1(y)}; \text{COMMIT}$$

$$T_2 \;:\; \text{BEGIN}; \underbrace{v := db_x}_{r_2(x)}; \underbrace{db_x := v + 1}_{w_2(x)}; \underbrace{v := db_y}_{r_2(y)}; \underbrace{db_y := v - 1}_{w_2(y)}; \text{COMMIT}$$

(a) Give all possible schedules that are *conflict equivalent* with the schedule $T_1; T_2$. Explain your answer.

(b) Are the schedules of Question 3(a) *serializable*? Explain your answer.

Suppose initially $db_x = 5$ and $db_y = 10$, the database uses immediate-update pessimistic concurrency control with row-level locking, and both transactions run concurrently in isolation level "SERIALIZABLE". Because they run concurrently, it is not clear how the individual operations of both transactions are interleaved.

(c) What are all possible outcomes (in terms of values for db_x and db_y) after running these two transactions? Explain your answer.
(Tip: there are more than two possible outcomes!)

↓

abort?

Question 5: NoSQL Databases and HBase (10 points)

Your task is to design the database schema for an email application that uses Google's BigTable. The application should be able to host millions of users and billions of emails that consist of header and body information. The main operations performed on the database are

1. `store(accountId, email)`
   which stores an incoming email for an account,
2. `list(accountId)`
   which lists the header information of the emails of an account, and
3. `retrieve(emailId)`
   which retrieves the email body by its id.

Answer the following question.

(a) Give the tables, column families and columns that you would use in the schema of this application. Explain your answer by giving reasons why this schema leads to an efficient application.

Question 6: 2-Phase Commit (10 points)

Suppose we run a distributed transaction on three servers A, B, and C. A transaction manager running on server M is the coordinator for the 2-Phase commit protocol.

Suppose M started the 2-Phase Commit protocol, it has sent out the prepare messages to the three cohorts, they all received it, they all sent back their votes to commit, the coordinator received the votes to commit messages from A and B, but not from C, and then M crashed.

C: aborted
A,B uncertain
M rollback

(a) Suppose it takes some time for the coordinator to come back on-line. Explain what actions the time-out protocol prescribes for a *cohort* to take when it time-outs waiting for a response from the coordinator.

(b) Explain what the coordinator would find in its log after the crash.

(c) Explain what actions the restart protocol prescribes for the coordinator to take right after it is brought back on-line again.

(d) What is the end result of these events: global commit or global abort? Explain your answer.