UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

BEng Honours Degree in Computing Part I
MEng Honours Degrees in Computing Part I
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C130

DATABASES 1

Wednesday 8 May 2002, 14:00
Duration: 90 minutes
(Reading time 5 minutes)

Answer THREE questions

Paper contains 4 questions
Calculators not required

Section A *(Use a separate answer book for this Section)*

1a Consider relation schema R(A,W,X,Y), S(B,W,X,Z).

i) What is the schema (i.e., the attributes) of R join S ?
Express R join S using only the operators product, select, project.

ii) What is the schema of R ÷ S[W,X] ?
What does this expression represent (in words)?

b Consider the following relation schema.

TimeData(<u>EmpId</u>, <u>ProjId</u>, Hours)

Project(<u>ProjId</u>, Manager, Location, Budget)

Attribute Manager in relation Project has the same domain as attribute EmpId in relation TimeData. ProjId in TimeData is a foreign key referring to ProjId in Project.

A tuple (e,p,h) in TimeData indicates that employee with identifier e has worked for h hours on project p. A tuple (p,m,l,b) in Project indicates that project p has a manager m, is located at location l, and has a total budget b.

Formulate the following queries in relational algebra.

i) For every project, find the number of hours worked by the manager on that project. Your query should return triples (manager, project, hours).

ii) Find the employees who work on every project located in York.

iii) Find the managers of the projects with the largest budgets.
You can assume that budget values can be compared using the comparison operator >.
Your query should return tuples (manager, project).
For this query only, give a *brief* explanation of how your algebra expression produces the desired answer.

c On the basis of the relation schema of part b formulate queries b(i) and b(ii) in *tuple relational calculus*.

The three parts carry, respectively, 25%, 50%, 25% of the marks.

2a Let R be the set of attributes of some relation R and let F be a set of functional dependencies on R.

i) Let $X \subseteq R$ be a set of attributes of relation R. Define X^+_F, the *closure* of attributes X under the functional dependencies F.

ii) Let $K \subseteq R$ be a set of attributes of relation R. Express in terms of the closure of K under F what it means to say that K is a *candidate key* of R.

iii) Show using the definition of part ii) that AB and BC are both candidate keys of the relation R(A,B,C,D,E) with the following functional dependencies:

AB -> E	C -> A
AD -> C	E -> D

b i) Define Boyce-Codd normal form.

ii) Show that the relation R in part a(iii) is not in Boyce-Codd normal form. You may assume without proof that there are no other candidate keys.

iii) The following are two BCNF decompositions of the relation R in part a(iii).

R[<u>C</u>,A] join R[<u>E</u>,D] join R[<u>B,C</u>,E]

R[<u>A,D</u>,C] join R[<u>E</u>,D] join R[<u>A,B</u>,E]

Explain carefully why both of these decomposition are lossless (non-loss). There is no need to define the term lossless (non-loss). You may state any standard results without proof.

c i) Define what is meant by a *dependency preserving* decomposition.

ii) Are either of the decompositions in part b(iii) dependency preserving? You will need to determine the projections of the functional dependencies F on the decomposed relations, and hence determine which of the decompositions are dependency preserving.

d i) Define what is meant by a multi-valued dependency A ->> B.

ii) Explain carefully why any relation that satisfies the functional dependency A -> B also satisfies the multi-valued dependency A ->> B.

iii) Are the relations in part b(iii) in Fourth Normal Form (4NF)? Justify your answer.

The four parts carry, respectively, 25%, 25%, 25%, 25% of the marks.

Section B *(Use a separate answer book for this Section)*

3a Explain what is meant by a *history* in the context of concurrency control.

Distinguish between a *serial* history and a *serialisable* history.

b Transactions T_1 and T_2 (given below) have been defined for a banking environment. In each transaction the operations included involve subtracting an amount from one account (e.g. x) and adding it to another account (e.g. y) as well as reading/writing the current/updated balances of the relevant accounts.

$T_1 : r_1 [x]\ w_1 [x]\ r_1 [y]\ w_1 [y]$

$T_2 : r_2 [y]\ w_2 [y]\ r_2 [x]\ w_2 [x]$

The following history has been suggested for the concurrent running of these two transactions:

$H = r_1 [x]\ w_1 [x]\ r_2 [y]\ w_2[y]\ r_1 [y]\ w_1 [y]\ r_2[x]\ w_2[x]\ c_1\ c_2$

i) By drawing the appropriate serialisation graph, show that this history is non-serialisable.

ii) Discuss why this history may nevertheless be considered to be *correct* (i.e produces consistent results).

iii) Give an alternative history that is non-serialisable and incorrect.

c Explain what is meant by the *two phase locking* (2PL) protocol.

Distinguish between *Basic* 2PL and *Strict* 2PL.

d The execution of a history is said to be *strict* if both $r_i [x]$ and $w_i [x]$ operations are delayed until all transactions Tj ($j \neq i$) that had previously written x have either committed or aborted.

Let history H represent an execution in which

$w_i [x] < o_j [x]$ (i.e. write precedes operation o)

where $o_j [x]$ is either $r_j [x]$ or $w_j [x]$.

Show that if this history is produced by a Strict 2PL scheduler, then the ensuing execution will be strict.

The four parts carry, respectively, 15%, 35%, 20%, 30% of the marks.

4a Explain what is meant by a *victim* in the context of deadlock recovery.

It is generally agreed that when choosing a victim one attempts to find the *least cost* solution. Explain what is meant by this and discuss how an appropriate victim that meets this criterion is chosen.

b Discuss what is meant by *livelock.*

Compare and contrast the *Wound-Wait* and *Wait-Die* approaches to deadlock prevention and show how livelock is prevented under these approaches.

c Explain what is meant by *transaction atomicity.*

In a distributed environment, in which several remote sites are connected by a network, it is possible for the execution of a transaction to be so arranged that different parts of the transaction are executed at separate sites.

Discuss the problem of ensuring overall transaction atomicity in such an environment and suggest a method whereby atomicity can be guaranteed.

The three parts carry, respectively, 30%, 35% and 35% of the marks.