UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

BEng Honours Degree in Computing Part III
MSc in Computing for Industry
MEng Honours Degree in Information Systems Engineering Part IV
BSc Honours Degree in Mathematics and Computer Science Part III
MSci Honours Degree in Mathematics and Computer Science Part III
MSc in Advanced Computing
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
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C312=I4.4

ADVANCED DATABASES

Tuesday 7 May 2002, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

SECTION A

1a Briefly describe, with an example using the histories notation, what is meant by *lost update*.

b In the context of two-phase locking (2PL)

i) The following is an *incorrect* implementation of 2PL. Determine how the execution fails to meet the general rules of 2PL.
$H_e = rl_1[o_1], r_1[o_1], wl_1[o_1], w_1[o_1], r_1[o_2], wu_1[o_1], wl_1[o_2], w_1[o_2], c_1, wu_1[o_2]$

ii) Rewrite H_e from (i) to obey strict 2PL

c The table (below left) shows the Recovery Manager log found when a database is executing a recovery, which includes a cache consistent checkpoint record. State which REDO and UNDO actions the recovery manager will execute, and restore the branch table (below right) to a consistent state.

LOG	b_4
UNDO	$w_4[b_{67}$, cash=29770.00]
REDO	$w_4[b_{67}$, cash=34005.25]
LOG	b_1
UNDO	$w_1[b_{56}$, cash=94340.45]
REDO	$w_1[b_{56}$, cash=84340.45]
LOG	b_2
UNDO	$w_2[b_{34}$, cash=10900.67]
REDO	$w_2[b_{34}$, cash=8900.67]
LOG	c_4
LOG	$cp_{1,2}$
UNDO	$w_2[b_{67}$, cash=34005.00]
REDO	$w_2[b_{67}$, cash=36005.25]
LOG	b_7
LOG	c_2
UNDO	$w_1[b_{34}$, cash=8900.67]
REDO	$w_1[b_{34}$, cash=18900.67]
UNDO	$w_7[b_{67}$, cash=36005.25]
REDO	$w_7[b_{67}$, cash=37005.25]

branch	
sortcode	cash
56	84340.45
34	8900.67
67	34005.00
77	40000.00

d Briefly describe why conflict serialisability is more tractable in terms of computer processing than view serialisability.

The four parts carry, respectively, 20%, 30%, 30% and 20% of the marks

2a Using the histories notation, show how the two transactions below could be ordered so that they are deadlocked. Show the waits-for graph for this deadlock state.

```
BEGIN TRANSACTION Booking
  UPDATE plane
  SET seat='reserved'
  WHERE seat_id=100;

  UPDATE credit_card
  SET bal=bal-@ticket_price
  WHERE CCowner_id=500;
COMMIT TRANSACTION Booking
```

```
BEGIN TRANSACTION Cancel_booking
  UPDATE credit_card
  SET bal=bal+@ticket_price
  WHERE CCowner_id=500;

  UPDATE plane
  SET seat='unreserved'
  WHERE seat_id=100;
COMMIT TRANSACTION Cancel_booking
```

b Using the histories notation, how would you order the transactions from (a) to ensure there is no deadlock?

c Briefly describe the role of the Buffer Manager in the context of DBMS engine architecture.

d Consider the following scenario in a DBMS buffer management system. If a tuple is in the main-memory it takes 20 ns to access it. If it is in the buffer and not in main-memory it takes 60 ns to load it into the main-memory. If the tuple is not in the buffer, 12 ms are required to fetch the tuple from the disk, followed by 60 ns to copy it to main-memory. The main-memory hit-rate is 0.9 (i.e. there is a 90% chance that the data can be found there) and buffer hit-rate is 0.6.

What is the average time in ns required to access a tuple on this system? (Explain your answer step-by-step).

The four parts carry, respectively, 30%, 20%, 25% and 25% of the marks

SECTION B

3a The following questions refer to the relations listed below. In the relations key attributes are underlined.

staff	
name	salary
Fred	1000
James	2100
Alex	0

pay		
name	payment	number
Fred	900	1
James	2000	1
Fred	900	2
Fred	900	3
James	2100	4
Fred	1000	4

i) The following SQL statement is to be executed on the tables above.

```
SELECT salary, payment
FROM staff,pay
WHERE staff.name=pay.name
AND number>3
AND staff.name<'J'
```

Translate the SQL to an equivalent relational algebra query tree, and then give an *optimised* distributed query tree over $pay_1, pay_2, staff_1, staff_2$ to execute on site S_3 in a distributed database, with the relations distributed between sites S_1 and S_2 as below. The optimisation should minimise the data sent by S_1 and S_2 to S_3.

$pay_1 = \sigma_{number<2} pay \qquad pay_2 = \sigma_{number \geq 2} pay$

$staff_1 = \sigma_{name<\text{'}J\text{'}} staff \qquad staff_2 = \sigma_{name \geq \text{'}J\text{'}} staff$

ii) The SQL query below is being executed on a distributed database where S_1 now holds staff and S_2 now holds pay.

```
SELECT staff.name, salary, payment, number
FROM staff,payment
WHERE staff.name=pay.name
AND salary>1500
```

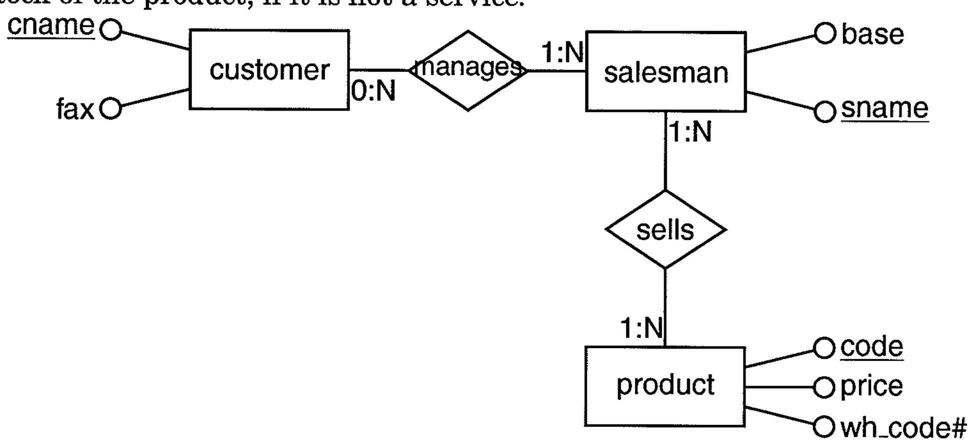
If the name attribute occupies 10 bytes of data, and all other fields occupy 4 bytes of data, compute the amount of data (in bytes) exchanged between servers when S_1 executes the query using the *direct join* approach, and when using the *semi-join* approach.

b Two export database schemas **sales** and **warehouse** are described below, which you should integrate into a single federated ER model. Apart from the final ER model, your answer should clearly enumerate the transformations applied during the integration, using notation such as

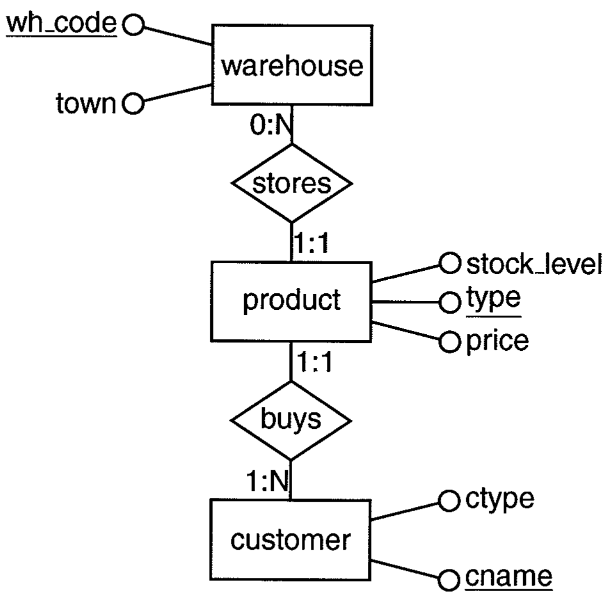
addEntity(⟨⟨silly⟩⟩, {X | salesman(X) ∧
salesman_manages_customer(X, Y) ∧ customer_fax(Y, *'123456'*)})

Your answer need *not* include the various expandEntity, expandAttribute, ... steps required at the end of schema conforming.

Sales: The database shown in the ER model below holds information about the sales force, which deals *only* with business customers, and not the general public. Salesman are allocated to products, and the range of products the company sells includes service contracts as well as the goods it manufactures. Each product is identified by a code AAXXXX where A is a letter and X a digit, we record its selling price, and the optional attribute wh_code records the code of warehouse where we hold stock of the product, if it is not a service.



Warehouse: The database shown in the ER model below holds details of the location of all products which take the form of physical goods that the company manufactures. Each product is identified by a type attribute AAXXXX where A is a letter and X a digit, and we record its buying price and stock_level. Products are related to the warehouse where they are stocked and the customers who purchase the products. Records are kept of *all* customers, which are classified by the ctype attribute being *'BN'* if a business customer or *'PR'* if a private customer.



The two parts carry, respectively, 36%, and 64% of the marks

4a The following shows the content of a database and two transactions that are run on the database.

person			
id	name	dept	salary
100	austin	computing	36000
101	stephen	computing	34000
102	waqaas	sales	30000
103	brian	sales	33300
104	david	purchase	28000

```
BEGIN TRANSACTION T1
    SELECT SUM(salary)
    FROM person
END TRANSACTION
```

```
BEGIN TRANSACTION T2
    UPDATE person SET dept='computing'
    WHERE id=104

    UPDATE person SET dept='purchase'
    WHERE id=100
END TRANSACTION
```

You are told that data has been fragmented, such that rows with ids 100 to 102 are on server S_1, and the remainder are on S_2.

i) Name what type of fragmentation has been used to distribute the data between S_1 and S_2.

ii) A concurrent execution of T1 and T2 has deadlocked as follows:
$r_2(p_{104}), w_2(p_{104}), r_1(p_{100}), r_1(p_{101}), r_1(p_{102}), r_1(p_{103}), r_2(p_{100}), \langle\text{deadlock}\rangle$
Draw the distributed waits-for graph for this deadlocked state.

iii) If most transactions take the general form of T1 or T2, how would you suggest redistributing the data between S_1 and S_2 to avoid creating a distributed waits-for graph.

iv) A new table pay(id,date,amount) has been added to the database, which record various payments made to persons, and for which id is a foreign key pointing at the person table. Give a relational algebra definition which would ensure that rows in pay are held on the same server as the person to which they pertain, and name the type of fragmentation you are using.

v) The following is an incorrect fragmentation of the person table. What two things are incorrect?

$person_1 = \pi_{name,dept} person$

$person_2 = \pi_{dept,salary} person$

b The following shows a temporal database, represented in the temporal structure, for the valid time records of a table recording the staff in a company, and a table recording the projects on which those staff work.

$t = 0$

staff	
name	dept
peter	cs
alex	maths
mike	cs

project	
name	title
peter	tempora
mike	cal
alex	int

$t = 1$

staff	
name	dept
peter	cs
alex	cs
mark	maths

project	
name	title
peter	hod
alex	hod

$t = 2$

staff	
name	dept
peter	cs
alex	cs
mark	maths
mike	cs

project	
name	title
peter	hod
mike	automed
mark	diff

For each of the following temporal relation algebra queries, list the output relation if the query is run at $t = 3$, and suggest what the query is intended to do (*e.g.* 'Finds all people who have left the company'). Note that ◆ is the *sometime in the past* operator, ■ is the *always in the past* operator, and $\overset{S}{\times}$ is the *since product* operator.

i) $\blacksquare \pi_{name}$ staff

ii) $\pi_{title} \blacklozenge$(project $\bowtie \sigma_{dept=\text{'}cs\text{'}}$ staff)

iii) (π_{name} project) $\overset{S}{\times}$ ($\sigma_{name=\text{'}alex\text{'} \wedge dept=\text{'}maths\text{'}}$ staff)

The two parts carry, respectively, 50%, and 50% of the marks