UNIVERSITY OF LONDON

291 0205

BSc EXAMINATION 2006
for External Students

COMPUTING AND INFORMATION SYSTEMS

CIS205 Databases

Duration: 3 hours

Date and Time: Tuesday 9 May 2006: 10.00 - 1.00pm

This paper is divided into two parts. Candidates should attempt THREE questions from Part A and THREE questions from Part B. All questions carry equal marks and full marks can be obtained for complete answers to SIX questions. Questions involving a description or explanation should, wherever possible, be accompanied by an appropriate example.

WRITE YOUR ANSWERS TO SECTION A AND SECTION B IN SEPARATE ANSWER BOOKS

Electronic calculators may be used. The make and model should be specified on the script. The calculator must not be programmed prior to the examination. Calculators which display graphics, text or algebraic equations are not allowed.

THIS EXAMINATION PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

PART A

1. (a) What is meant by saying that a sorting method "sorts a list within itself"? Why is it important that a sorting method should be able to sort a list within itself?

[2]

(b) Outline the Quicksort method for sorting a list. Your description should show how Quicksort would be implemented as a recursive procedure. Briefly give a justification of why the complexity of Quicksort is O(nlog(n)).

[7]

(c) In the context of external sorting what are partitions? How, in general, do the sizes of partitions affect the efficiency of the sorting algorithm?

[4]

(d) Describe fully the method of sorting using replacement selection.

[6]

(e) External sorting methods can have a complexity which is the same as the average complexity of Quicksort and better than Quicksort's worst case. Why, therefore do we not use external sorting methods in every case?

[3]

(f) Describe an external sorting method which may use Quicksort as an essential component.

[3]

2. (a) In the context of file storage, what would be the primary reason for using hashing?

[1]

(b) A commonly used technique for implementing hashing is to divide storage into "sectors" or "buckets". Describe fully a hashing technique, which uses buckets. Your description should contain an account of methods for the insertion, retrieval and deletion of records. You should also discuss the advantages and disadvantages of using buckets and considerations on the size and number of buckets.

[12]

(c) A student identity number is composed of a letter followed by nine digits.

The first letter is either "U", "M" or "P" to indicate whether they are undergraduate, masters or doctorate students.

The next two digits are the year of entry to the university.

The third digit indicates the faculty in which they are registered; there are five faculties in the university.

The fourth digit will be 1, 2 or 3 depending on the level at which the student entered the degree programme.

The remaining five digits are allocated in order starting from 00000 as students register. Each year, when registration restarts, the first student will be given the number 00000.

Describe a possible hashing algorithm that uses the identity number for allocating student records into a set of buckets. Carefully explain any considerations you would take into account in designing your hashing algorithm. In particular, describe the problems of primary and secondary clustering and justify why your hashing method should not be susceptible to these forms of clustering.

[12]

3. (a) What is meant by an indexed file?

[2]

(b) A file index may be described as either static or dynamic. What is the difference between a static and a dynamic index? Under what circumstances would you consider using each type of index?

[6]

(c) Outline the structure of either a static index or a dynamic index. Describe how the index you have described may handle the insertion and deletion of records.

[10]

(d) The "B" in "B-tree" stands for "balanced". Why is a B-tree so called? How does a B^+-tree differ from a B-tree? What are the advantages and disadvantages of using a B^+-tree in preference to a B-tree?

[7]

4. (a) A supermarket maintains an electronic catalogue of all the products that it sells. For each product, there is a record that contains the following information.

(i) A unique number which corresponds to the barcode on the product label.

(ii) A single keyword for the manufacturer of the product.

(iii) A short description of the item.

(iv) The location of the item in the supermarket.

(v) The current price of the product.

(vi) The number of items of that product that are currently in stock.

This information is also printed on a card that is physically attached to the shelf where the product is displayed.

The above information is kept in one file. Records will need to be retrieved from the file in several ways and for several purposes. The system must be capable of the following functions.

(i) All the details in the record must be retrieved very quickly following the input of the product barcode number.

(ii) When the price or location of the item is changed, the record will need to be updated and the shelf card for the item reprinted.

(iii) When an item is sold or a delivery of new stock arrives, then the record must be updated.

(iv) A list of all products from a particular manufacturer can be produced.

(v) A list of all items in stock but below a given number can be produced daily.

Describe a suitable record structure and procedures that will allow all the above operations to be carried out. (Note, the information is held as one record per product and in just one file.) Also outline what procedures would be required for the insertion and deletion of items from the file.

[15]

(b) One night, the computer disc on which all of the above information is stored develops a fault and is unusable the following morning. Describe fully a practical back-up and recovery procedure that would enable the supermarket to be operational again very quickly.

[10]

PART B

5. A direct-sales company has a large database of its customers, where a 'customer' is defined as anyone who has ever ordered anything from the company. The main customer relation consists of about 1.5 million tuples, growing at a rate of about 15% annually. Each tuple occupies about 2 kilobytes of disc space.

The relation includes the following attributes, among others:

CUSTOMER-CODE
CUSTOMER-FIRST-NAME
CUSTOMER-MIDDLE-NAME
CUSTOMER-FAMILY-NAME
CUSTOMER-TITLE
CUSTOMER-ADDRESS-1
CUSTOMER-ADDRESS-2
CUSTOMER-ADDRESS-3
CITY
POST-CODE
CREDIT-CARD-TYPE
CREDIT-CARD-NUMBER
EXPIRY-DATE
DELIVERY-ADDRESS-1

DELIVERY-NOTES
TELEPHONE-NUMBER-1
TELEPHONE-NUMBER-2
FAX-NUMBER

DATE-OF-FIRST-ORDER
DATE-OF-LAST-ORDER
NUMBER-OF-ORDERS
TOTAL-SPENT
INQUIRY-NOTES

There are also several dozen fields holding data from an extensive questionnaire which most customers send in (in return for a free gift) used by the Marketing Department to research sales trends. The customer's sex, age-group, income-level, family size and composition, favourite entertainments and hobbies, and other buying information, are all held here. These fields occupy several hundred bytes.

Of course the chief use of the main customer relation is to process customer orders, to add new customers, to change customer details when necessary, and to delete customers who have moved leaving no forwarding address.

In addition to the above procedures, the company uses the main customer relation in the following ways:

(1) To post its 300-page annual catalogue to all 'active customers'. An 'active customer' is defined as one who has ordered anything within the last 24 months. (About half of the active customers place orders two or three times a year.) The catalogue is expensive to produce and to post, so only 'active customers' receive it. About 10% of the names on the database are 'active customers', who place about 80% of the orders. The other 20% of orders come from new customers, or 'inactive customers'. (Because customers who have not ordered anything within the last 24 months may still order something, they are not deleted from the relation.)

(2) To post an abbreviated 'catalogue highlights and previews' brochure to inactive customers. This brochure is relatively cheap to produce and to post, and it has proved worth it to post it to inactive customers, as some will become 'active customers' again after receiving this brochure.

(3) For research into sales trends among active customers. For example, a few weeks after running a television advertising campaign aimed at getting new customers, the marketing department will access the main customer relation to see how many new orders have been placed, and from what type of customer. Much of this work consists of one-off *ad hoc* queries by researchers in the Marketing Department. For example, a researcher might want to see how many new customers have been added to the database since a certain date, broken down by region and by sex. (This will necessarily involve a

complete scan of the relation.) Note that queries from the Market Research Department only ever reference the POST-CODE, TOTAL-SPENT and market research fields.

About 80% of orders are by telephone, and the remainder are by post. When an existing customer rings up with an order, his or her information held in the main customer relation must be accessed quickly. However, the time taken to access the main customer relation has been increasing, and is becoming unacceptably long. It is also the case that the Marketing Department would like to be able to get faster response to its **ad hoc** (on-line, one-off) queries, which take place at the same time (and therefore compete with) order processing accesses to the main relation.

A. Propose **two** changes (which may be independent of each other) in the relational design of this database which might result in faster processing of orders and *ad hoc* queries from the Marketing Department. (All relevant fields are already indexed. Hardware upgrades are ruled out.) Briefly explain **why** each change could result in faster processing, and mention any possible **disadvantages** of each change.

[12]

B. A colleague notes that the main customer relation is not fully normalized. He points out that the POST-CODE attribute determines the CITY attribute, and proposes that the relation should be split into two normalized relations: one, retaining all the attributes of the current relation except for CITY, and a second, consisting of POST-CODE and CITY (where POST-CODE would be the key). Briefly discuss a possible **disadvantage** of doing this *in the context of this particular database and the way it is used.*

[1]

C. The Ruritanian Army wishes to record, for each of its five million soldiers, the types of weapons for which that soldier has passed a proficiency test. (Almost all of its soldiers have passed a proficiency test for at least two types of weapon. At the moment there are five weapon types, but new types may be added in future.) Two designs are proposed:

Design I. The first design has one tuple for each soldier, with the soldier's unique SERIAL-NUM as key, and a set of attributes of type Boolean, with one attribute per weapon type, as in the following example.

SERIAL-NUM	RIFLE	PISTOL	GRENADE	MACHINE-GUN	MORTAR
A0001	Y	Y	Y	N	N
A0002	Y	N	N	N	N
A0003	Y	N	N	Y	N
.	.	.	.	.	.

Design II. The second design has only two attributes, SERIAL-NUM and WEAPON, making up a composite key, as in the following example.

SERIAL-NUM	WEAPON
A0001	Rifle
A0001	Pistol
A0001	Grenade
A0002	Rifle
A0003	Rifle
A0003	Machine gun
.	.

(Question continues on next page)

Assume that neither relation will be indexed, that attributes of type Boolean take up one byte, and that no data compression or token-substitution methods will be used on attributes of type CHARACTER-STRING, such as the attribute WEAPON in Design II. (In other words, the field width for WEAPON in Design II will at least as large as the largest string which is a value in this field.)

Briefly evaluate each design, with an explanation of your reasoning, from the point of view of:

(1) The amount of storage space each relation is likely to occupy.

(2) The amount of processing time each relation is liable to require when being accessed in response to the following queries:

   (a) List all soldiers who have passed their proficiency test for both Rifle and Machine-gun.

   (b) List all weapon types for which soldier A0002 has **not** yet passed his proficiency test.

(3) The amount of processing time each relation is liable to require when it is necessary to add information about soldiers' proficiencies in a **new** type of weapon, such as a flame-thrower. **[12]**

6.

A. When a database designer has finished designing the set of tables which will hold the data for a proposed database, he or she must decide, for each table, which attributes to **index**.

Briefly discuss the effect, in terms of

(1) total database size,
(2) performance of the system with respect to queries (which do not alter the database),
(3) performance of the system with respect to updates (which do alter the database)

of indexing a field (attribute) of a particular table.

Your answer should include both (a) the **effect** and (b) a *short* **explanation** for the effect you are describing in each case.

[7]

B. A key concept of relational database design is **normalisation**.

(1) What is it? (Give an example.)
(2) What are arguments in its favour? (Refer to your example in your previous answer.)
(3) What considerations might lead us to decide that we do **not** want to fully normalize a particular relation? (Give an example.)

[8]

C. Last year a small company which sells semi-finished mechanical parts to other companies implemented an order processing system, based around a well-known microcomputer relational database package. The data in the system is held in the following 'base relations': customer details; product details; employee details. There are also relations with order details (with attributes ORDER-NUM, PRODUCT-NUM, QUANTITY); a relation linking customers and orders (with attributes CUST-CODE, ORDER-NUM); and a relation linking salesmen and customers (called **CUSTOMER-CONTACT**, with attributes EMPNUM, CUST-CODE).

The work was done in a hurry by someone who was not very experienced in using DBMS packages. An adequate, but very basic system was implemented. Not all of the features of the DBMS package were used. Some problems have revealed themselves as the system has been used.

Some of the problems are:

(1) Impossible data values sometimes get entered when new tuples are added to the database. For example, although all telephone numbers have seven digits, data entry errors when adding a new customer sometimes result in telephone numbers with 6 or 8 digits, thus making it necessary to look the customer up in the telephone book or even write to him in order to get his correct phone number. The company director thinks that there should be some way of dealing with this, but doesn't know enough about computers to suggest how.

(2) One of the relations in the database (**CUSTOMER-CONTACT**) links each regular customer to a specific salesman, who is responsible for handling that customer's account. This may include periodically telephoning the customer and informing him of special offers the company is making. Occasionally a customer goes bankrupt, or is marked as a bad debt. In this case the salesman with responsibility for that customer is supposed to be notified to drop him from his list, and the tuple linking that salesman to that customer is manually deleted from **CUSTOMER-CONTACT**. But from time to time this deletion is overlooked,

(Question continues on next page)

and salesmen have found themselves contacting 'deleted' customers, leading to confusion and embarrassment. The company director wants to know if this deletion can be done automatically, when the customer is deleted from the main customer relation.

(3) To handle customer inquiries over the phone, the order expediter often calls up company details on her computer, by accessing the main customer relation. At the moment, all fields relating to one customer are displayed on one line (since there is one record, or tuple, per company), and sometimes relevant fields are off-screen, forcing the expediter to scroll across to find them. Is there any way that these details could be arranged on the screen in a more accessible way?

(4) The relation linking customers and orders has three fields: CUSTOMER-NUMBER, ORDER-NUMBER, and DATE-PLACED. The key was declared by the original designer as CUSTOMER-NUMBER + ORDER-NUMBER. Occasionally, a data entry error has allowed two different customers to be linked to the identical order, although this never happens in practice. (Two or more customers never place a joint order.) Is there any way that this error can be prevented?

(5) There have been cases where a customer order has been recorded for 1000 units of a particular item, even though no-one ever orders more than 100 of that item. The company has a list of all its products (each identified by a unique product number), along with, for each one, a 'maximum normal order quantity', which clerks now check by eye, as they did under the previous manual system. But the company would like to make this automatic, if possible.

For each of these problems, **briefly** describe how the facilities of a modern DBMS package might be used to deal with the problem. Be as specific and concrete as possible.

[10]

7.

A. As part of its employee-motivation scheme, a company holds information about which sports its employees enjoy, and about certain company-approved resorts which offer facilities for different kinds of sports. (Make the normal 'closed world' assumption about this database: that is, if a given employee is not listed as enjoying a particular sport, you may assume that he or she does not enjoy it; and if a given resort is not listed as offering a particular sport, you may assume that it definitely does not offer it.) Employee preferences are held in the relation **ES**, shown here with a sample of tuples.

ES

KEY: EMPLOYEE+SPORT
MEANING: The given EMPLOYEE *enjoys* the given SPORT.

EMPLOYEE	SPORT
E123	Surfing
E123	Swimming
E123	Tennis
E234	Swimming
E345	Swimming
E345	Tennis

Resorts which the company has approved for sponsored employee holidays, and the sports for which they cater, are shown in the relation **SR**.

SR

KEY: SPORT + RESORT
MEANING: the given SPORT *is offered by* the given RESORT.

SPORT	RESORT
Riding	Happy Holidays
Riding	The Fun Ranch
Surfing	Happy Holidays
Surfing	Sea Adventures
Surfing	Beachfront Joy
Swimming	Happy Holidays
Swimming	Sea Adventures
Swimming	The Fun Ranch
Tennis	Happy Holidays

Consider the following relational algebra operations on the relations **ES**, **SR** and relations derived from them. (In these operations, *RESTRICT* has the same meaning as *SELECT* in older textbooks. The *RESTRICT* of a relation yields a new relation with the same attributes as the old relation, but with only those tuples specified by the accompanying expression. If we *RESTRICT* **ES** *WHERE* EMPLOYEE = E345, we get a new relation with the same attributes as **ES**, but with only those tuples where the EMPLOYEE number is E345. If we then *PROJECT* this new relation *OVER* SPORT, we get a relation whose *meaning* is all SPORTs enjoyed by employee E345. Note that in SQL, the operations of *RESTRICT* and *PROJECT* are combined in the expression

(Question continues on next page)

SELECT: using SQL, we would find all SPORTs enjoyed by E345 with the expression SELECT SPORT FROM **ES** WHERE EMPLOYEE=E345.)

R1 ← *PROJECT* **ES** *OVER* EMPLOYEE

R2 ← *RESTRICT* **ES** *WHERE* SPORT = Surfing

R3 ← *PROJECT* **R2** *OVER* EMPLOYEE

R4 ← *RESTRICT* **ES** *WHERE* SPORT = Tennis

R5 ← *PROJECT* **R4** *OVER* EMPLOYEE

R6 ← *UNION* (**R3**, **R5**)

R7 ← *INTERSECT* (**R3**, **R5**)

R8 ← *DIFFERENCE* (**R3**, **R5**)

R9 ← *RESTRICT* **ES** *WHERE* SPORT <> Surfing

R10 ← *PROJECT* **R9** *OVER* EMPLOYEE

R11 ← *DIFFERENCE* (**R1**, **R3**)

R12 ← *RESTRICT* **SR** *WHERE* RESORT = The Fun Ranch

R13 ← *JOIN* **R12**, **ES** *ON* SPORT

R14 ← *PROJECT* **R13** *OVER* EMPLOYEE

R15 ← *DIFFERENCE* (**R1**, **R10**)

Give the **meanings** of

(1) R1
(2) R3
(3) R6
(4) R7
(5) R8
(6) R10
(7) R11
(8) R14
(9) R15

[14]

(Question continues on next page)

7.(cont.) B. In the following relations, passport numbers are unique for each individual, as are employee numbers. Surnames are not necessarily unique. A department can have many employees working in it, but an employee will work in only one department. A department will have one manager, and an employee can manage only one department. A project can have several employees assigned to it, and an employee can be assigned to more than one project. All employees must retire by the time they are sixty-five years old; no person younger than eighteen can work for the company.

EMPLOYEES

KEY: EMPLOYEE-NUM

EMPLOYEE-NUM	SURNAME	BIRTHDATE	PASSPORT-NUM	DEPT.
E123	Chan	23.08.70	P987	Accounting
E234	Ismail	12.02.75	P876	Marketing
E345	Jones	15.11.64	P765	Accounting

DEPARTMENTS

KEY: NAME

NAME	MANAGER
Accounting	E345
Marketing	E567
Finance	E678

PROJECTS

KEY: PROJECT+MEMBER

PROJECT	MEMBER	DATE-ASSIGNED
P1	E789	06.10.95
P1	E890	08.10.95
P2	E789	23.08.96

Using examples drawn from the above relations where possible, define any **five** of the following terms:

(1) Candidate Key
(2) Primary Key
(3) Foreign Key
(4) Composite Key
(5) Referential Integrity
(6) Entity Integrity
(7) Attribute Integrity

[5]

(Question continues on next page)

7.(cont.) **C.** Consider the following relation, which contains information on horses (names and birthdates), the races they have entered in (date, and result), and the tracks they have raced on (the name of the track, and the city that the track is located in).

RACE-RESULTS

KEY: HORSE + RACE-DATE

HORSE	RACE-DATE	BIRTHDATE	TRACK	CITY	RESULT
West Wind	12.07.04	01.01.00	Hippodrome	Leeds	3rd
Man O'War	12.07.04	01.01.99	Hippodrome	Leeds	1st
West Wind	26.07.04	01.01.00	Concourse	London	2nd
Lucky Jim	26.07.04	01.01.98	Royal Court	London	1st
Man O'War	26.07.04	01.01.99	Royal Court	London	2nd
Lucky Jim	02.08.04	01.01.98	Concourse	London	1st
West Wind	14.09.04	01.01.00	Hippodrome	Leeds	2nd
Pindar	14.09.04	01.01.98	Hippodrome	Leeds	1st

The Functional Dependencies in this relation are:

HORSE+RACE-DATE → TRACK A horse will be raced at most once in a day.

HORSE+RACE-DATE → RESULT

HORSE → BIRTHDATE

TRACK → CITY No two cities have tracks with the same name. A city can have more than two tracks.

Turn this relation into a set of fully normalized relations. Do **not** create more relations than are necessary for the process of normalization. Indicate the **key** of each relation.

[6]

8. In a certain country, houses and flats are sold in the following manner. A **seller** wishing to sell a **property** registers this fact with the National Property Authority. (A seller may have more than one property to sell, but a given property will always be registered in the name of a principal seller, who will act for any other owners in subsequent negotiations.) The seller then instructs the National Property Authority to list his property with one or more **estate agencies**. (Estate agencies are privately-owned businesses which undertake to sell properties: they compete with each other in terms of commission taken, methods of advertising property, and so forth.) When a property is listed with a particular agency this is called a *listing*. Each agency employs several **estate agents** (but an agent works for only one agency). A **prospective buyer** *registers* with one or more estate agencies, and tells the agency in general terms what kind of property he or she is looking for. The agency shows the prospective buyer leaflets describing likely properties which may be of interest. If a buyer is interested in a particular property, an appointment for a *showing* may be made, in which a particular estate agent arranges to show a particular property to a prospective buyer at a certain date. A buyer may, of course, want to see a property more than once. If a prospective buyer wants to buy a property, he makes an *offer*, via a particular estate agent, specifying the amount he is willing to pay for a specific property. A prospective buyer may make offers for more than one property, and two or more prospective buyers may make offers for the same property, but a prospective buyer can only make one offer (of a given amount) for a particular property, and can only make an offer for a particular property with one estate agent.

A. Draw an Entity/Relationship diagram showing the *relationships* among **Sellers**, **Properties**, **Estate Agencies**, **Estate Agents**, and **Prospective Buyers**.

[19]

B. Assume **Properties** are uniquely identified by PNUMs, **Agencies** by ACODEs, **Estate Agents** by ENUMs, and both **Prospective Buyers** and **Sellers** by CUST-IDs. Design a set of normalized relations which could hold the information shown in your Entity/Relationship diagram in part **A** of this question. Do **not** add any attributes which are **not** mentioned in the description above, but **do** include any attributes which **are** mentioned. Do **not** bother to put in relations which hold information about just one entity type: only show relations which show how two or more entity types are related to each other.

[6]

END OF EXAMINATION