ECM2419

(with Answers)

UNIVERSITY OF EXETER

COLLEGE OF ENGINEERING, MATHEMATICS AND PHYSICAL SCIENCES

COMPUTER SCIENCE

Examination, January 2020

Database Theory and Design

Module Leader: Dr Hongping Cai

Duration: TWO HOURS

Answer ALL questions.

The marks for this module are calculated from 70% of the percentage mark for this paper plus 30% of the percentage mark for associated coursework.

No electronic calculators of any sort are to be used during the course of this examination.

This is a CLOSED BOOK examination.

Question 1

(a) There are a number of important functions provided by the *database management system*. Describe *four* of them.

(8 marks)

[BOOKWORK]

Any 4 of the following functions are correct. **[2]** each.

- *Data storage management*. Besides storage of data, it is also involved with a database's efficiency in relation to storage and access speed. Users do not need to know how data is stored or manipulated.
- *Data access and application programming interfaces*. SQL is the most common query language supported by the majority of DBMS vendors. It allows users to define the database, usually through a data definition language (DDL). It also allows users to insert, update, delete and retrieve data from the database, usually througha data manipulation language (DML).
- *A system catalog*. A system catalog stored the description of data items and is accessible to users.
- *Data communication interfaces*. Users' requests for database access are transmitted to DBMS in the form of communication messages.
- *Integrity services*. It ensures that both the data in the database and changes to the data follow certain rules.
- *Transaction management*. It ensures that all the updates corresponding to a given transaction are made or none of them is made.
- *Concurrency control services*. It ensures that the database is updated correctly when multiple users are updating the database concurrently.
- *Backup and recovery services*. It ensures that the database could be recovered in the event that the database is damaged in an way.
- *Authorization/security management*. It protects the database against unauthorized access, either accidental or intentional.

(b) There are a number of *computer-based security mechanisms* for protecting the database from threats. List *four* of them.

(4 marks)

> Computer-based security mechanisms include authentication, access control, views, integrity, encryption, audit, backup & recovery and RAID technology. (Any 4 are correct, **[1]** each)

(c) What is *deadlock*? Which *three main methods* could we use to deal with deadlock?

(8 marks)

> **[BOOKWORK]**
>
> Deadlock is an impasse that may result when two (or more) transactions are each waiting for locks to be released that are held by the other. **[2]**
>
> The three main methods to deal with deadlock is:
>
> - Timeout. A transaction will wait for a (database defined) period to acquire a lock. If this time runs out then the whole transaction is rolled back and restarted. **[2]**
> - Deadlock prevention. Transactions are ordered using transaction timestamps. The two algorithms are wait-die algorithm and wound-wait algorithm. **[2]**
> - Deadlock detection and recovery. Once deadlock has been detected (with wait-for graph), the DBMS needs to abort one or more of the transactions. **[2]**

(d) (i) What does it mean that a schedule is *serializable*?

(2 marks)

> If a set of transactions excecutes concurrently, we say that the (nonserial) schedule is correct if it produces the same results as some serial execution. Such a schedule is called *serializable*.

(ii) Consider three transactions T_1, T_2 and T_3, and one schedule S given below. Draw the *precedence graphs* for this schedule, and state whether it is *serializable* or not.

T_1: $r_1(X); r_1(Z); w_1(X);$

T_2: $r_2(Z); r_2(Y); w_2(Z); w_2(Y);$

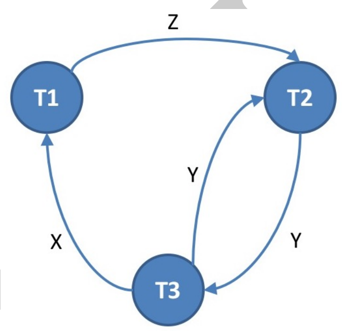
T_3: $r_3(X); r_3(Y); w_3(Y);$

S: $r_1(X); r_2(Z); r_3(X); r_1(Z); r_2(Y); r_3(Y); w_1(X); w_2(Z); w_2(Y); w_3(Y);$

Where $r_i(Z)$ and $w_i(Z)$ indicate a read and a write operation respectively by the i-th transaction on data item Z.

(7 marks)

The schedule's precedence graph **[4]** is as follows:



As there are 2 cycles ($T_1 \to T_2 \to T_3 \to T_1$ **[1]**, $T_2 \to T_3 \to T_2$ **[1]**), S_2 is a not serializable. **[1]**

(e) What are *checkpoints*? Why are they important in database recovery?

(5 marks)

[BOOKWORK]

Checkpoints are the point of synchronization between the database and the transaction log file. At a checkpoint, all buffers are force-written to secondary storage. **[2]**

In the event of failure, we check the log file to find the last checkpoint. **[1]** The checkpoint record identifies which transactions need to be redone. **[1]** It limit the amount of searching and subsequent processing. **[1]**

(f) Desribe two desirable characteristics of a *distributed DBMS*.

(6 marks)

One characteristic is distribution transparency, **[1]** which means that users perceive the database as a single, logical entity. **[1]**

The other characteristic is local atonomy. **[1]** Local data is locally owned and managed. **[1]** Each local database operates independently. **[1]** Local systems are able to operate effectively even if remote databases are not available. **[1]**

(Total 40 marks)

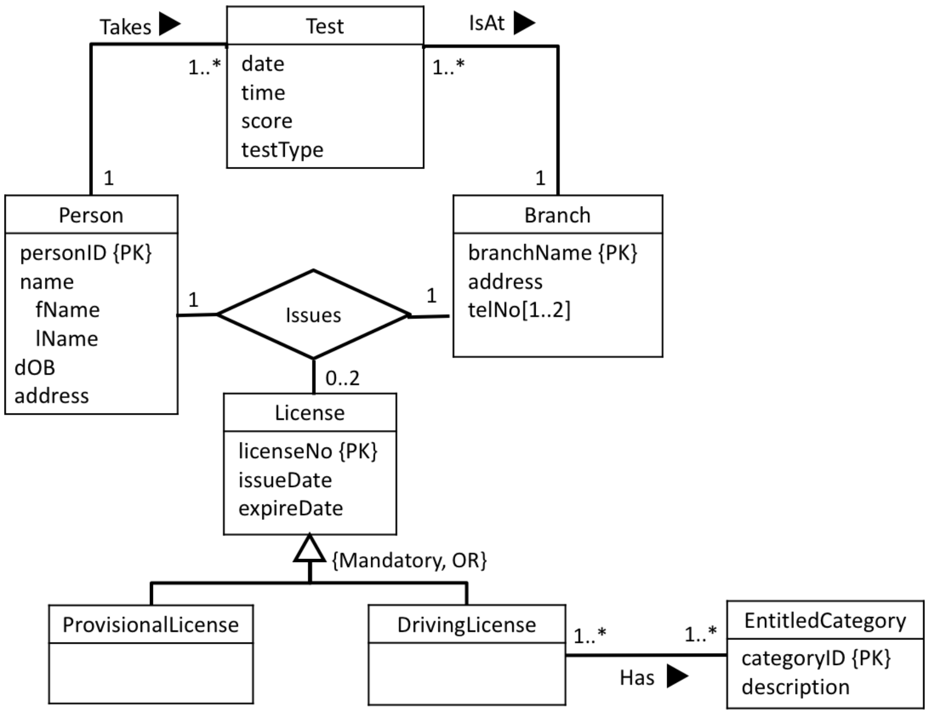
Question 2

(a) Motor vehicle branches manage driving tests and issue driving licenses.

- Any person who wants a driving license must first take a driving theory test at any Motor Vehicle Branch. Each person has a unique ID, name (first name and last name are recorded separately), date of birth and address recorded. Each branch has a unique name, address and one or two telephone numbers.
- If a person fails the theory test (recorded with the date, time and score), they can take the test again any time at any branch.
- Once a person passes the theory test, they are issued a *provisional* driving license with the issue date, expiry date and a unique license number.
- The person may take the practical test at any branch any time before the expiry date of the *provisional* driving license. Similar to the theory test, the practical test also has the date, time and score recorded.
- If the person passes the practical test, the branch issues a *full* driving license with recorded information similar to that of the *provisional* license. The only difference is that a *full* driving license also records the entitlement categories that specify what vehicles can be driven. For each category, a unique category ID and a description of the entitled vehicles are specified.

Draw an *entity-relationship diagram* (using UML notation) with *specialisation/generalisation* concepts for this motor vehicle management system as described above. Identify all entities, relationships, attributes (including primary keys) and multiplicity constraints.

(17 marks)

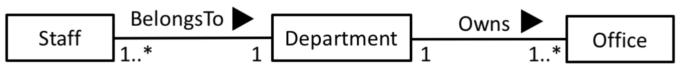


Marks:

- 7 entities with their attributes **(7 Marks)** 'telNo[1..2]' could be replaced with two attributes (tel1, tel2).
- 4 primary keys in the 4 entities **(2 Marks)**
- Relationship 'Issues', with multiplicity **(2 Marks)**
- Relationship 'Takes' and "IsAt", with multiplicity. **(3 Marks)** Alternatively, this relationship could also be represented as a ternary relationship. It could also has two subclasses (TheoryTest and PracticalTest)
- Relationship 'Has', with multiplicity **(1 Marks)**
- Superclass/subclass with {Mandatory, OR} **(2 Marks)**

(b) Give an example of a *fan trap* in ER modelling.

(3 marks)



Any diagram where two or more 1: relationships fan out from the same entity is correct.*

(c) Farmer Lucy owns a herd of dairy cows. In a spreadsheet she records the volume of milk produced each day by each of the cows. Each cow is required by law to have a unique ear tag number so that its ownership can be tracked over its whole life. The following table shows a small subset of her data, which is unnormalized.

CowName	EarTagRef	BirthDate	Date	MilkYield	Date	MilkYield	Date	MilkYield	Date	MilkYield	Date	MilkYield
Daisy	D201001	03/05/2010	01/10/2012	32	02/10/2012	34	03/10/2012	12	04/10/2012	38	05/10/2012	31
Ermintrude	D201002	12/05/2010	01/10/2012	28	02/10/2012	29	03/10/2012	30	04/10/2012	27	05/10/2012	28
Mabel	D201003	14/05/2010	03/10/2012	36	04/10/2012	38	05/10/2012	37	06/10/2012	38	07/10/2012	36

(i) What are the two main objectives of normalisation in database design?

(3 marks)

One objective is to remove redundancy;**[1]** the other is to remove potential insertion, modification, deletion anomalies.**[2]**

(ii) Briefly describe the three steps required to transform an unnormalized table into the *3rd normal form (3NF)*.

(3 marks)

First remove repeating groups to transform the table into the first normal form (1NF) **[1]**. Then remove partial dependencies to transform the table into the second normal form (2NF) **[1]**. Then remove transitive dependencies to transform the table into the third normal form (3NF) **[1]**.

(iii) Transform the above table into relations in the *3rd Normal Form (3NF)* by using relational schemas (with primary keys underlined) rather than detailed tables.

(4 marks)

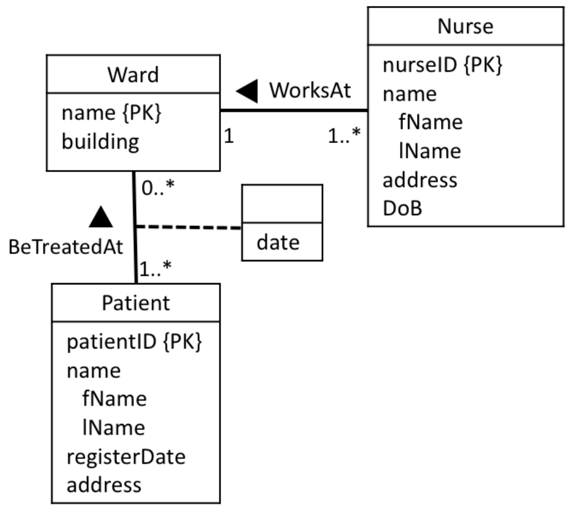
Cow (<u>EarTagRef</u>, CowName, BirthDate) **[2]**

MilkYields (<u>EarTagRef</u>, <u>Date</u>, Yield) **[2]**

(Total 30 marks)

Question 3

(a) Consider the following entity-relationship diagram.



Transform the above *ER diagram into a relational model*, reporting primary keys and foreign keys.

(8 marks)

Nurse (nurseID, fName, lName, address, DoB, wardName)

Primary Key: nurseID **[1]**

Foreign Key: wardName references Ward(name) **[1]**

Ward(name, building)

Primary Key: name **[1]**

Patient(patientID, fName, lName, registerDate, address)

Primary Key: patientID **[1]**

Treatment(wardName, patientID, date) **[1]**

Primary Key: wardName, patientID, date **[1]**

Foreign Key: wardName references Ward(Name) **[1]**

Foreign Key: patientID references Patient(patientID) **[1]**

(b) Consider the following relations from a Police Traffic database.

Vehicle (vehicleID, make, model, colour, personID)

Person (personID, name, address, NIN, licenseNumber, expiryDate)

Fine (vehicleID, date, time, amount, offenceID)

Offense (offenceID, description, maximumFine)

The **Fine** relation records fines for traffic offences, while the **Offense** relation records different types of offenses that may be committed.

(i) Write the following queries in *relational algebra*. To shorten the expression, you may use initials letters in place of the full relation names, e.g., O stands for 'Offense'.

A) Find all the offenses whose maximum fine is more than £100. Display the description and maximum fine.

(3 marks)

$$\pi_{\text{description, maximumFine}}(\sigma_{\text{maximumFine>100}} O)$$

B) List the date and amount of every fine charged to Floyd Pepper since 1 January 2019. (Note: The date is represented by '2019-01-01')

(4 marks)

$$\pi_{\text{date,amount}}(\sigma_{\text{name='Floyd Pepper'} \wedge \text{date>='2019-01-01'}}(P \bowtie V \bowtie F))$$

(ii) Write the following queries in *SQL statements*.

A) Display the make and model of the vehicle with vehicleID 'RX63LDN'.

(3 marks)

SELECT make, model

FROM Vehicle

> **WHERE** vehicleID = 'RX63LDN';
>
> *Marks allocation*: SELECT **[1]**, FROM **[1]**, WHERE **[1]**

B) Display the car registration number (vehicleID) for all cars which have accumulated more than £1,000 of fines.

(4 marks)

> **SELECT** vehicleID
>
> **FROM** Fine
>
> **GROUP BY** vehicleID
>
> **HAVING** SUM(amount)>1000;
>
> *Marks allocation*: SELECT-FROM **[1]**, GROUP BY **[1]**, HAVING **[1]**, SUM() **[1]**

C) List the largest fine for each type of offense in London area, i.e., for which the person involved in has an address containing "London".

(5 marks)

> **SELECT** f.offenceID, **MAX**(f.amount)
>
> **FROM** People p, Fine f, Vehicle v
>
> **WHERE** p.address **LIKE** '%London%'
>
> **AND** p.personID = v.personID
>
> **AND** v.vehicleID = f.vehicleID
>
> **GROUP BY** f.offenceID;
>
> *OR:*
>
> **SELECT** f.offenceID, **MAX**(f.amount)
>
> **FROM** People p
>
> **WHERE** p.address **LIKE** '%London%'
>
> **INNER JOIN** Vehicle v **ON** p.personID = v.personID
>
> **INNER JOIN** Fine f **ON** v.vehicleID = f.vehicleID

GROUP BY f.offenceID;

Marks allocation: SELECT-FROM-MAX **[2]**, WHERE or INNER JOIN ... ON **[1]**, LIKE **[1]**, GROUP BY **[1]**

D) Renew Floyd Pepper's driving license, with the new expiry data 2021-01-20. (Assume Floyd Pepper is a unique name in the database)

(3 marks)

UPDATE Person

SET expiryDate='2021-01-20'

WHERE name='Floyd Pepper';

Marks allocation: UPDATE **[1]**, SET **[1]**, WHERE **[1]**

(Total 30 marks)