



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Examination Answer Books

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

Complete, fold and seal
(see note 2 below)

SURNAME: Davies

FORENAME(S): Daniell Wyn

1	Student Number	00179762
2	Title of Examination Paper	Database Management Systems
3	Module Code	ACCA 5014
4	Date and Time of Examination	09/05/2019 9:05 AM
5	Examination Venue	G42
6	Seat Number	
7	Number of Answer Books handed in	1

[illegible]

(Gweler yr ochr arall am y fersiwn Cymraeg)

1a.i CREATE TABLE TutorBooking(
 Booking Ref int Not Null,
 Booking Date Datetime CHECK (Booking Date >= CurrDate)
 TutorNO int Not Null,
 StudentNO int Not Null,
 SubjectID Varchar (6) Not Null,
 Subject Level int,
 Hourly Rate Decimal DEFAULT '1.00';
)

1a.ii Alter Table TutorBooking ADD CONSTRAINT PK_BookingRef
 PRIMARY KEY (Booking Ref);

1a.iii Alter Table TutorBooking ADD CONSTRAINT FK_TutorNO
 FOREIGN KEY TutorNO REFERENCES Tutor(TutorID);
 Alter Table TutorBooking Add constraint- FK_StudentNO
 Foreign Key StudentNO References Student(StudentNO);
 Alter Table TutorBooking Add constraint- PK_SubjectID
 Primary Key SubjectID References Subject(SubjectID);

1a.iv. Create Table TutorSubject(
 TutorNO int Primary Key Not Null,
 SubjectID int Primary Key Not Null,
 KSLLevel Varchar (10),
 ALevel Varchar (10),
 HLevel Varchar (10),
 Min Hourly Rate Decimal - Check (Min Hourly Rate < Max Hourly Rate)
 Max Hourly Rate Decimal;
)

1b.i Select - TutorNO, - Firstname, - Surname, - Email, - PhoneNO
 From Tutor
 Where DBS Expiry >= 01-06-2019
 Order By DBS Expiry

15.ii. Select (*)

from Tutor

Where DBS ExpiryDate < Getdate

insert values (status) 'Pending DBS Check'

15.iii. Select min (Hourly rate)

from Tutor Booking

Select max (Hourly rate)

from Tutor Booking

Select Avg (Hourly rate)

from Tutor Booking.

Q2a. External Schema refers to how a database is ~~split~~ split into separated views which are made specifically for each user or application. The Conceptual ~~Schema~~ Schema relates to the entirety of the database as those with access to the conceptual schema will be able to view all relations and constraints within a database as it is not split by views. Internal Schema refers to the method in which a database is stored physically so this schema refers to storage of the database.

Q2b. An index is used in order to speed up queries performed by a database by reducing the amount of records and fields that are searched through by indexing columns of a table that are frequently used in queries.

2bii. Analysing transactions before implementing indexing is important as creating indexes can result in an extra cost to the amount of storage needed for an index. If an index is made for a less important query that is rarely run then it would be a large cost in storage capacity to index a table that is rarely used. The amount of space required to index also is determined by table size so the larger the table the larger the index will be. However this would speed up queries on the

larger table as indexing would ensure that the queries could only look for the field(s) that are contained in the index rather than the entire relation which could be very faster. I would look for instances where the same fields are being called frequently and consider making ~~an~~ index on the more used ~~fields~~ fields when queries are run.

5 1/2
✓

- 23iii With regards to indexing the Tutor table I would consider creating an index based on the Firstname, Surname, DBS Expiry Date and Status field. as you could perform checks on such the expiry dates and status of each tutor DBS requirement whilst indicating the tutor's full name. To create the index you would use the following.

Create Index idx-DBScheck ON Tutor (Firstname, Surname, ~~DBS Expiry Date~~ DBS Expiry Date, Status);

- 23iv. I would consider creating a secondary index out of the City column as it would keep a list of where each tutor operates out of that can be freely called ~~the~~ when from querying to find where each tutor is located.

2 ✓

- 23v. Insert Into TutorBooking (StudentID, SubjectID, TutorID, BookingRef, Booking Date) Values

('25', 100, 66, 500, '30-05-2019 10:00:00')
(25, 100, 66, 500, '31-05-2019 10:00:00');

3 1/2 ✓
19

- Q4 ACID refers to the set of properties that transactions should have to ensure a consistent state. Atomicity refers to how a transaction must either fully succeed or fully fail, so if any part of a transaction fails the whole thing is a failure. Consistency refers to how a transaction must leave the database in a consistent state before and after its ~~execution~~ execution if the database was consistent before the transaction and it must be in the same state after.

Q4. Con- Isolation refers to how the system should treat each transaction as if they are the ^{only} transaction being executed. No transaction should be able to affect the existence of changes directly. Durability refers to how each transaction, when committed, should be stored in the database through faults or downtime. If the database fails and recovers, then it should update the data once it is back online.

Q4b. Consistency Control ensures that no transactions can overwrite another whilst they are both running. This is done using either lock-based protocols or timestamp based protocols to determine which transaction can read or write to a data item at a given time.

Q4c. A lost update occurs when one transaction begins to alter data but before it can commit the change another transaction begins doing the same resulting in the second transaction reading the original value not the first transaction's value, meaning that the second transaction commits overwriting the T1 value resulting in the update being lost ^{from} ~~from~~ transaction 1. Uncommitted dependency occurs when two or more transactions are run together but one transaction reads from another transaction's value which is rolled back as it is incorrect, but the second transaction retains this value as it has not rolled back. Inconsistent analysis is when two or more transactions perform calculations at the same time but one transaction influences another calculation either directly or indirectly resulting in a calculation where the values used cannot be traced back.

Q4(i)	Time	T1	T2	Salx(500)
	H		begin	500
	E2	begin	read sal	500
	E3	read salx	salx = salx + 100	500
	E4	salx = salx - 50	write salx	600
	E5	...	commit	600
	E6	write salx		450
	E7	commit		450

The last update has occurred in this instance as one transaction has read from the original value of the salx record rather than the T2 transaction committed salx. This could have been prevented by running T1 after T2 has finished as the correct value would have been committed and used ~~correctly~~ correctly by T1 resulting in the final salx being 550.

Q4(ii) T3 would need to be redone as it is unclear if there was any commits before or after the crash so it would need to be redone.

Q4(iii) T4 would need to be undone as it happened after the crash and may have used data from T3 which should have updated previous.

Q4(iv) T1 is ~~unaffected~~ not affected by the crash as it occurred before the crash happened.

T2 is also unaffected as it was committed to the database previous.

$$\begin{array}{r}
 19 \\
 19 \\
 \hline
 21 \frac{1}{2} \\
 19 \\
 \hline
 40 \frac{1}{2}
 \end{array}$$

$$79\%$$