



12 Pages

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 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.

8. Every book must be handed in. In no circumstances may answer books, used or unused,

7	Number of Answer Books handed in	
6	Seat Number	16
5	Examination Venue	
4	Date and Time of Examination	09/05/19 9:30 AM
3	Module Code	ACCAS014
2	Title of Examination Paper	DATABASE MANAGEMENT SYSTEMS
1	Student Number	88924

[illegible]

Complete, fold and seal
(see note 2 below)

SURNAME: SUTTON

FORENAME(S): BRYN

a1)

CREATE TABLE TutorBooking ✓

BookingRef

CHAR(5) NOT NULL, ✓

BookingDate

DATE TIME NOT NULL, ✓

TutorNo

INT(5) NOT NULL, ✓

StudentNo

INT(5) NOT NULL, ✓

SubjectId

INT(5) NOT NULL, ✓

SubjectLevel

INT(2) NOT NULL, CHECK (SubjectLevel > 0),

HourlyRate

Decimal (3,2) NOT NULL ✓

a11)

ALTER TABLE TutorBooking

ADD CONSTRAINT [PK-TutorBooking]

PRIMARY KEY (BookingRef); ✓

a111)

ALTER TABLE TutorBooking ✓

ADD CONSTRAINT [FK-TutorBooking-Tutor]

FOREIGN KEY (TutorNo) REFERENCES Tutor (TutorNo);

ADD CONSTRAINT [FK-TutorBooking-Subject]

FOREIGN KEY (SubjectId) REFERENCES Subject

(SubjectId); ✓

2

1

6

aiv) CREATE TABLE TutorSubject ✓

TutorNo
 NOT NULL CONSTRAINT
 [FK-TutorSubject-Tutor]
 FOREIGN KEY REFERENCES
 Tutor (TutorNo),
 SubjectId
 NOT NULL CONSTRAINT
 [FK-TutorSubject-Subject]
 FOREIGN KEY REFERENCES
 Subject (SubjectId),
 ✓

KSLevel
 VARCHAR(20),
 ✓
 ALLevel
 VARCHAR(20),
 ✓
 HELevel
 VARCHAR(20),
 ✓
 MinHourlyRate DECIMAL(3,2),
 MaxHourlyRate DECIMAL(3,2),
 ✓

CONSTRAINT [PK-TutorSubject-Tutor-Subject]
 PRIMARY KEY (TutorNo, SubjectId) REFERENCES
 Tutor, Subject (TutorNo, SubjectId)
 ✓

);

b)

SELECT TutorNo, ~~Name~~ First Name, Surname,
 Email, PhoneNo
 FROM Tutor
 WHERE DBS Expiry Date BEFORE '01-JUL-2019'
 ORDER BY DBS Expiry Date, Surname, First Name
 HSD

2 ✓

1 b ii)

UPDATE DBS EXPIRY DATE
FROM Tutor
WHERE DBS EXPIRY DATE = 'Expired'
TO DBS EXPIRY DATE = 'Pending'

2 a)

Question Two

1 b iii)

SET Average AVG (Average) X

2 b i)

The purpose of an index is to help increase the speed & priority of certain table columns. Typically indexes are created on tables and/or columns that are frequently used, in order to first track the selected process.

2 b ii)

It is important to analyse transactions before implementing indexes because every index ~~will~~ ^{could} be ~~be~~ CREATED, READ, UPDATED or DELETED, which therefore putting more strain on the database. unwanted indexing can affect the performance of the database. Therefore it is important that only highly utilised tables & columns are indexed for efficiency.

3

CLUSTERED

Examiner Only

25iii)

CREATE INDEX ~~Index~~

SELECT FirstName, Surname, DOB, Email, phone
FROM Tutor
ORDER BY Surname, FirstName, DOB;

This index can be used to quickly search the tutors without a primary key, however the combination of first & last name with the addition of DOB will be enough to identify them without the use of a primary key.

26iv) I would choose the phone number column as a secondary index as it is the only other single item unique to each tutor.

25v) ~~UPDATE TABLE~~

~~INSERT BookingRef, BookingDate, TutorNo, StudentNo, SubjectLevel, HourlyRate~~
~~INTO~~

INSERT INTO TutorBooking
SELECT BookingRef, BookingDate, TutorNo, StudentNo, SubjectLevel, HourlyRate
INTO ('500', '30-May-2019', '66', '25', '100', '10:00:00:000');

~~SELECT~~
SELECT BookingRef, BookingDate, TutorNo, StudentNo, SubjectLevel, HourlyRate
INTO ('500', '31-May-2019', '10:00:00:000', '66', '25', '100');

9

3a)

1) Un Authenticated login, this allows any one to access a database and make changes. This could lead to loss of personal information i.e. emails, addresses or financial information.

2) Unrestricted privileges to all users. In a large company a database access should be restricted to various sections and tiers of the company. Not doing so could lead to loss of private information and or tampering with the database.

3b)

3) SQL Injection attacks, if the database is part of an online service then it could be prone to hackers who can get access from a browser and change the server side code. Whitelisting or allowing only ^{trusted} safe/commands to be given outside of the admin privileges can help step this.

1 Authentication is used as confirmation users attempting to log into a database have the authority to do so. Authentication can be double layered having both server & database authentication. In addition authentication allows admin to audit who has been accessing what. ~~meta~~ increasing the level of security.

3

Question 3

Question No.

Examiner Only

[illegible]

3b11)

Access controls & user privileges are used to allow or restrict certain user types from accessing or changing ~~data~~ data-base objects. This often occurs by allowing or restricting control of the CRUD functions. For example 'Guests' may only be allowed to Read a table, while 'Staff' could Read & update and 'Admin' ~~can~~ have access to all.

3

3b11)

Views are able to show sections of a table, however they can be created using horizontal and/or vertical views, which only show selected parts of a table. For example on an employee view their name & Job title may be present but their salary will not. In addition views can be restricted to those having the correct level of permissions. This greatly improves security as it specifies who can see what & how much they can see. ✓

3

3C)

CREATE VIEW TutorListSubject AS
 SELECT First Name, Surname, Email, City,
~~Join~~ FROM Tutor
 Join Tutor, TutorSubject

CREATE VIEW TutorListSubject AS ✓
 SELECT First Name, Surname, Email, City, KSLLevel
 ALevel, MLevel, MinHourlyRate, MaxHourlyRate,
 FROM Tutor, TutorSubject
 Join [Tutor-TutorSubject]
 WHERE Status = 'Active' ✓
 GROUP BY Tutor, TutorSubject;

5

20

