**Q1.**

A veterinary practice with four different surgeries intends to use a relational database box to store the data that it needs to manage its business.

Customers of the practice are pet owners who bring their pets to one of the surgeries for appointments. The surgeries are staffed by vets.

•   Each customer is identified by a unique identity number and the customer’s forename, surname and telephone number are recorded.

•   Each pet is identified by a unique identity number and the pet’s name, type and date of birth are recorded.

•   Each surgery is identified uniquely by its name. The town in which it is located and the surgery’s telephone number are recorded.

•   Each vet is identified by a unique identity number and the vet’s forename and surname are recorded.

A pet is owned by one or more customers and each customer may own any number of pets. Over their lifetimes, pets may attend many appointments.

To make an appointment for a pet, a customer contacts a surgery. The appointment is made for the pet to take place on a particular date and time at a specific surgery.

Each vet is associated with one surgery which they work at; each surgery is staffed by several vets.

(a)  Complete the entity-relationship diagram below for a **fully normalised** relational database to store the data required by the veterinary practice.

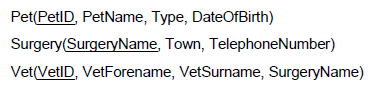
Some of the entities and relationships have been drawn for you. You need to draw the remaining **three** entities and clearly show the relationships between the entities and their degree.



**(3)**

(b)  Develop a **fully normalised design** for a relational database to store the information required by the veterinary practice. To help you, the Pet, Surgery and Vet relations have already been defined in **Figure 1**.

**Figure 1**

****

Using the format shown in the information above list all the other relations that will need to be created, together with the attributes that each will contain.

Underline the attribute(s) that will form the entity identifier (primary key) in each relation.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

(c)  The SQL query in **Figure 2** has been written to produce a list of all of the vets who work at the surgery in the town of Torquay. Some errors have been made in the query.

**Figure 2**

****

Describe **two** errors that have been made in the query. You should not give the omission of a semi-colon (;) as one of the errors.

Error 1 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Error 2 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(d)  The database is stored at the practice’s head office. Staff at the individual surgeries access it using a client-server database system, which enables the management of concurrent access to the database.

Describe an example of a problem that could occur if no system were in place to manage concurrent access to the database.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

**(Total 12 marks)**

**Q2.**

Two methods that can be used to manage concurrent access are:

•   record locks

•   timestamp ordering.

Select **one** of these methods and describe how it manages concurrent access.

Method selected: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

How it works \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(Total 2 marks)**

**Q3.**

Employees at a bank use client computers to access data that is stored on a database server.

The database server uses software to query and modify data stored in a database on hard disk drives. It returns the results of these queries to the clients over the bank’s computer network.

The performance of the system is unsatisfactory: the time-delay between a client sending a query to the server and the client receiving the results is unacceptably long.

Explain how the performance of the system might be improved. You should consider the following factors that might be affecting the performance:

•   the hardware of the server

•   the design of the computer network

•   the database and software running on the server.

In your answer you will be assessed on your ability to follow a line of reasoning to produce a coherent, relevant and structured response.

**(Total 12 marks)**

**Q4.**

Athletes, who are members of teams, compete in running events, which are held at fixtures throughout the year.

For example, athlete 15 might compete in the Girls’ 1500m Under 18 race in the fixture at Marsten on 12 September 2018.

A relational database is used to store the details of which athletes enter each event at each fixture. The relations used in the database are shown in **Figure 1**.

**Figure 1**

Athlete(AthleteID, Surname, Forename, DateOfBirth, Gender, TeamName)

EventType(EventTypeID, Gender, Distance, AgeGroup)

Fixture(FixtureID, FixtureDate, LocationName)

EventAtFixture(FixtureID, EventTypeID)

EventEntry(FixtureID, EventTypeID, AthleteID)

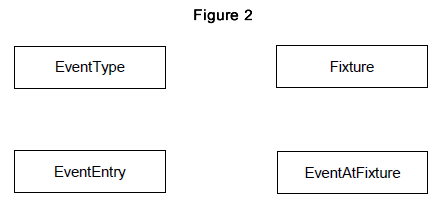
•   Each Athlete, EventType and Fixture is identified by a unique identity number, for example AthleteID for athletes.

•   An EventType is a type of event, such as Boys’ 100m Under 15 race.

•   If an athlete wants to take part in an event at a particular fixture, then an entry is created in the EventEntry relation to represent this.

(a)  **Figure 2** shows an incomplete entity-relationship diagram for part of the database.

Draw lines on **Figure 2** to show the degree of any **three** relationships that exist between the four entities shown.



**(2)**

(b)  **Figure 3** shows an SQL statement that is intended to make a table to represent the Athlete relation. The statement contains some errors.

**Figure 3**

       CREATE TABLE Athlete (

          PRIMARY KEY AthleteID,

          VARCHAR(50) Surname,

          VARCHAR(30) Forename,

          DATE DateOfBirth,

          VARCHAR(6) Gender,

          VARCHAR(30) TeamName

       )

You may assume that all of the data types used in **Figure 3** are valid and the field lengths are appropriate.

State **two** errors that have been made.

**Error 1:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Error 2:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)  State **two** reasons why database designs, such as this one, are usually normalised.

**Reason 1:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Reason 2:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

**Figure 1** is repeated below.

**Figure 1 (repeated)**

Athlete(AthleteID, Surname, Forename, DateOfBirth, Gender, TeamName)

EventType(EventTypeID, Gender, Distance, AgeGroup)

Fixture(FixtureID, FixtureDate, LocationName)

EventAtFixture(FixtureID, EventTypeID)

EventEntry(FixtureID, EventTypeID, AthleteID)

A list is to be produced of the names of all athletes who are competing in the fixture that is taking place on 17/09/18. The list must include the Surname, Forename and DateOfBirth of these athletes and no other details. The list should be presented in alphabetical order by Surname.

(d)  Write an SQL query to produce the list.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

**(Total 11 marks)**

**Q5.**

A garage services and repairs cars. It uses a relational database to keep track of the jobs that customers have booked for it to carry out. The database includes jobs that have been completed and jobs that are waiting to be done.

The details of the jobs that the garage does, together with the parts that it stocks and uses are stored in the database using the four relations shown in **Figure 1**.

**Figure 1**

|  |
| --- |
| Job (JobID, CarRegNo, JobDate, InGarage, JobDuration)  Car (CarRegNo, Make, Model, OwnerName, OwnerEmail, OwnerTelNo)  Part (PartID, Description, Price, QuantityInStock)  PartUsedForJob (JobID, PartID, QuantityUsed) |

•   Each car has a unique CarRegNo.

•   A type of car can be uniquely identified by the combination of its Make and Model. Different Makes may use the same Model name and a particular manufacturer (Make) will produce several different car Models.

•   A booking made for a car on a particular date counts as one job, regardless of how many different tasks are completed upon it.

•   A job might require the use of any number of parts, including zero.

•   Some of the details are stored in the database as soon as a booking is made and others are only added when a job has been completed.

The attribute JobID is the Entity Identifier (Primary Key) of the Job relation.

(a)     If the JobID attribute were not included in the Job relation, which other attribute or attributes that are currently in the relation could probably be used as an Entity Identifier (Primary Key) instead?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

It has been suggested that the owner details (OwnerName, OwnerEmail, OwnerTelNo) should not be stored in the Car relation and that a new relation should be created to store owner details separately from car details.

(b)     Explain why storing the owner details separately would improve the design of the database.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     On the incomplete Entity-Relationship diagram below show the degree of any **three** relationships that exist between the entities.

|  |  |  |
| --- | --- | --- |
| Job |  | Car |
|  |  |  |
|  |  |  |
| Part |  | PartUsedForJob |

**(2)**

When an appointment is made for a job, this is represented in the Job relation. At the time of booking, the InGarage attribute is set to False and the JobDuration attribute is set to 0:00. When the car arrives at the garage the value of the InGarage attribute is changed to True. When the job is finished the value of the JobDuration attribute is updated to indicate how long the job took and details of the parts used are recorded in the database.

The Job with JobID 206 has been completed. The job took 1 hour 30 minutes (1:30) and used two of the parts with PartID 12.

(d)     Write the SQL commands that are required to record the amount of time that the job took in the database.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

(e)     Write the SQL commands that are required to record in the database the fact that two of the parts with PartID 12 were used.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

**Figure 1** is repeated below.

|  |
| --- |
| Job (JobID, CarRegNo, JobDate, InGarage, JobDuration)  Car (CarRegNo, Make, Model, OwnerName, OwnerEmail, OwnerTelNo)  Part (PartID, Description, Price, QuantityInStock)  PartUsedForJob (JobID, PartID, QuantityUsed) |

A mechanic needs to produce a list of all of the parts used on the job with JobID 93 for a customer.

This list must include the PartID, Description, Price (each) and QuantityUsed of each part, and no other details. The parts in the list should be ordered by PartID with the parts with the lowest PartIDs nearest to the top of the list.

(f)      Write an SQL query to produce the list.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

There are restrictions on which parts can be fitted to which cars. For example:

•   The driver’s door mirror with PartID 104 can only be fitted to one particular make and model of car.

•   The ignition switch with PartID 27 can be fitted to any model of car for one particular make as the maker uses the same ignition switch in all models.

•   The tyre with PartID 97 can be fitted to a wide range of cars of different makes and models as it is a standard size.

If the information about which parts could be fitted to which makes and models of cars were represented in the database, it could be used to help a mechanic identify the correct parts to use for a job.

(g)     Explain how the database design could be modified to represent which makes and models of car a part can be fitted to.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

**(Total 18 marks)**

**Q6.**

A company is building an e-commerce website. The website will display details of the products that the company sells and allow customers to place orders. Customers must register on the website before they can place an order and each order can be for one or more different products.

The product, customer and order details will be stored in a relational database.

It was originally proposed that the following three relations were required:

   Product(ProductNumber, ProductPrice, ProductDescription, QuantityInStock)

   Order(OrderNumber, OrderDate, CustomerID, OrderingComputerIPAddress, ProductNumber, Quantity)

   Customer(CustomerID, CustomerName, Address, Postcode, EmailAddress, PaymentCardNumber)

The computer programmer identified a problem with the Order relation and stated that it should be divided up into two separate relations:

   Order(OrderNumber, OrderDate, CustomerID, OrderingComputerIPAddress)

   OrderLine(OrderNumber, ProductNumber, Quantity)

(a)     Describe the problem that the programmer identified with the original Order relation and explain what the cause of this problem was.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(b)     Complete the Entity-Relationship diagram below to show the degree of any **three** relationships that exist between the entities in the improved database design.

|  |  |  |
| --- | --- | --- |
| Customer |  | Product |
|  |  |  |
| Order |  | OrderLine |

**(3)**

A web page is required that will display a summary of the products that are on a particular order.

The summary must include only the ProductNumber, ProductDescription, ProductPrice and the Quantity of the product that has been ordered. These must be displayed in ascending order of ProductDescription.

(c)     Write an SQL query that will find the data needed to produce the order summary web page for order number 97.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

**(Total 10 marks)**

**Q7.**

A parcel delivery company uses a relational database to store information about the deliveries that it makes. These details include information about each customer who sends a parcel, the individual parcels being delivered and pricing details.

The company offers three different service speeds, which are "Express", "Standard" and "Economy". The price that is charged for delivering a parcel depends upon the service speed selected and the weight of the parcel (to the nearest gram). For each service speed, parcel prices are split into bands for a range of weights. For example, for the "Express" service, the price bands are as follows:

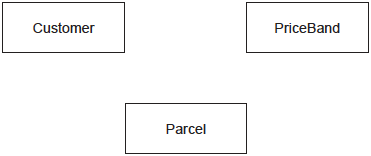
|  |  |  |
| --- | --- | --- |
| **Minimum Weight (g)** | **Maximum Weight (g)** | **Price** |
| 0 | 249 | £1.99 |
| 250 | 499 | £2.99 |
| 500 | 999 | £3.99 |
| 1000 | 4999 | £4.99 |
| 5000 | 19999 | £9.99 |

Similar price bands, but with different prices, exist for the "Standard" and "Economy" services.

The details are stored using the three relations in the figure.

|  |
| --- |
| Customer(CustomerID, Title, Forename, Surname) |
| PriceBand(ServiceSpeed, MinWeight, MaxWeight, Price) |
| Parcel(ParcelID, ServiceSpeed, Weight, DateSent, CustomerID, RecipientName,  HouseNumber, Street, Town, County, Postcode) |

(a)     On the incomplete Entity-Relationship diagram below, show the degree of the three relationships that exist between the entities.



**(2)**

(b)     The price that is charged for an "Express" delivery, weighing between 1000 and 4999 grams is to be increased to £5.99. Complete the SQL statements below to make this update.

UPDATE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SET \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WHERE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

(c)     Write a query that will list all of the parcels sent by the customer whose **CustomerID** is 109.

For each parcel, the list should include the **DateSent**, the **Postcode** that the parcel was sent to, the **ServiceSpeed** that was used and the **Price** charged, and no other details.

The list should be presented in order, with the parcel sent the longest time ago at the top of the list and the parcel sent most recently at the bottom.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(6)**

The figure is repeated below to help you answer part **(d)**.

|  |
| --- |
| Customer(CustomerID, Title, Forename, Surname) |
| PriceBand(ServiceSpeed, MinWeight, MaxWeight, Price) |
| Parcel(ParcelID, ServiceSpeed, Weight, DateSent, CustomerID, RecipientName,  HouseNumber, Street, Town, County, Postcode) |

(d)     The **Street**, **Town** and **County** parts of a recipient’s address can all be identified from the **Postcode**.

This means that the Parcel relation is not normalised and contains redundant data.

Redesign the Parcel relation, and create any new relations that you think are necessary, to eliminate this redundancy from the database to produce a normalised design.

Use the same notation that has been used in the figure when answering this question part. Make sure that you underline the attribute(s) that make up the primary key in each relation.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

**(Total 15 marks)**

**Q8.**

A school stores information about its sports day in a relational database.

The details of the track events are stored using the three relations in **Figure 1** .

|  |
| --- |
| **Figure 1** |
| Athlete (AthleteNumber, Forename, Surname, Class, Gender, DateOfBirth) |
| Race (RaceNumber, Gender, Distance, Type, StartTime) |
| RaceEntryAndResult (RaceNumber, AthleteNumber, TimeSet) |

Each athlete who takes part in a race is given a unique AthleteNumber. Athletes can run in more than one race. If they do, they keep the same AthleteNumber for the entire day.

Many races are run throughout the day. An example race would be the boys 80m hurdles, the third race of the day, which starts at 13:30. The entry in the Race table for this race is shown in the table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RaceNumber** | **Gender** | **Distance** | **Type** | **StartTime** |
| 3 | Boys | 80 | Hurdles | 13:30 |

When an athlete is entered into a race, a record of the entry is created in the RaceEntryAndResult table. Initially, the TimeSet is recorded as 00:00.00 (meaning 0 minutes, 0 seconds, 0 hundredths of a second) to indicate that the race has not yet been run. After the race has been run, if the athlete successfully completes it, then their TimeSet value is updated to record the time that they achieved in minutes, seconds and hundredths of a second. The TimeSet value remains at 00:00.00 for athletes who fail to complete the race.

The primary keys in the Athlete and Race relations have been identified in **Figure 1** by underlining them. The correct primary key for the RaceEntryAndResult relation has not been identified.

(a)     In **Figure 2** below, underline the appropriate attribute name(s) to identify the correct primary key for this relation.

|  |
| --- |
| **Figure 2** |
| RaceEntryAndResult (RaceNumber, AthleteNumber, TimeSet) |

**(1)**

(b)     Relations in a database should usually be fully normalised.

Define what it means for a database to be fully normalised.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     On the incomplete Entity-Relationship diagram below show the degree of the **three** relationships that exist between the entities.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Athlete | | | |  | | RaceEntryAndResult | | | |
|  |  |  |  |  |  |  |  |  |  |
|  | | | Race | | | |  | | |

**(2)**

(d)     Athlete number 27 is to be entered into race number 6.

Write the SQL commands that are required to make this entry.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(e)     **Figure 1** is repeated below.

|  |
| --- |
| **Figure 1 (repeated)** |
| Athlete (AthleteNumber, Forename, Surname, Class, Gender, DateOfBirth) |
| Race (RaceNumber, Gender, Distance, Type, StartTime) |
| RaceEntryAndResult (RaceNumber, AthleteNumber, TimeSet) |

Athlete number 27 sets a time of 0:18.76 (0 minutes, 18 seconds, 76 hundredths of a second) for race number 6.

Write the SQL commands that are required to update the athletes entry for this race, to store this time in the TimeSet field.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

(f)     The competition organisers want to produce a list of all of the athletes who took part in race number 6 with the athlete who won (set the lowest time) at the top and the other athletes below the winner in the order in which they finished.

Only athletes who finished the race should be included in the list.

The following information should appear for each athlete: AthleteNumber, Forename, Surname and TimeSet.

Write an SQL query to produce the list.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

(g)     The database system is to be extended for use in an inter-school athletics league. Users at any school in the county will be able to access the system to input the results of races.

It is possible that two users might try to access or update the system at the same time.

Explain the conditions under which simultaneous access to a database could cause a problem, and how this could be dealt with.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

**(Total 18 marks)**

**Q9.**

A school enters Year 12 and Year 13 students for AS and A Level qualifications.

Each qualification is identified uniquely by a combination of a subject name and level, e.g. ‘Computing’ and ‘A Level’. A qualification with the same name can exist at both AS Level and A Level.

Qualifications are split up into modules. Each module is identified by a code, e.g. ‘COMP3’. No two modules can have the same code, even if they are in different qualifications. A module also has a name and a number of UMS points associated with it.

Each student who is being entered for modules has their Forename, Surname, Centre Number and Candidate Number recorded. Taken together, the Centre Number and Candidate Number uniquely identify a student.

When a student is entered for a module their Centre Number, Candidate Number and the Exam Session that the entry is for (e.g. ’summer 2015’) are recorded, together with the information necessary to identify which module the entry is for. A student who is unhappy with their result can re-sit a module in a later session.

(a)     Develop a **normalised** design for a relational database to store the information described above.

List the names of **all** of the relations together with the attributes that each will contain.

Underline the attribute(s) that will form the primary key in each relation.

To help you, the Student relation has already been defined.

Student (CentreNumber, CandidateNumber, Forename, Surname)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

(b)     The primary key in the Student relation is made up of two attributes, as no one attribute can uniquely identify a student.

What name is given to this type of key?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

**(Total 6 marks)**

**Q11.**

A government agency is responsible for storing information about vehicles and their owners. Each vehicle that is driven must be registered with this agency. Vehicles must be insured to be driven, so the agency also keeps a record of vehicle insurance policies.

Details of the vehicles, owners and insurance policies are stored in a relational database using the following three relations:

Vehicle(RegistrationNumber, OwnerID, Manufacturer,  
Model, Colour, EngineSize, DateRegistered)

Owner(OwnerID, Title, Forename, Surname, HouseNumber, Street, Town, Postcode)

Insurance(PolicyNumber, RegistrationNumber, DateStarted, PolicyType, ExcessAmount)

In this system, the following restrictions apply to some attributes:

•        RegistrationNumber: a mixture of exactly 7 letters and numbers, eg MA11FXB

•        EngineSize: a whole number value representing the capacity of the engine, eg 1597

•        PolicyType: can be either ‘Comprehensive’ or ‘Third Party’ and nothing else

•        ExcessAmount: a monetary value, eg 100

(a)     Complete the following Data Definition Language (DDL) statement to create the Insurance table, including the key field.

CREATE TABLE Insurance (

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(3)**

(b)     The owner of the vehicle with registration number DF24JUT has had his car repainted so that its colour is now pink.

Complete this SQL statement to update the data in the Vehicle table to reflect this change.

UPDATE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SET \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WHERE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     A police officer is following a car with registration number AB72XHC. She wants to use the computerised system to check some details about the car and its owner.

Write an SQL query that could be used to retrieve the Model and Colour of the car and the Forename and Surname of the car’s owner.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

(d)     The police officer requests the information using a hand held terminal that connects to the Internet. She types the vehicle registration number into a form on a secure webpage and the details about the car and owner are then displayed in the web browser on the terminal.

A server-side script is used to search for the required information.

(i)      Explain what a server-side script is.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(ii)     The server-side script includes the statement:

RegNo = Request("RegistrationNumber")

Explain what this statement does when executed.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(iii)    The server-side script includes the statement:

Response.Write("Owner is " + Forename + " " + Surname)

Explain what this statement does when executed.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

|  |
| --- |
| The definitions of the three relations in the database are repeated here.  Vehicle(RegistrationNumber, OwnerID, Manufacturer, Model, Colour, EngineSize, DateRegistered)  Owner(OwnerID, Title, Forename, Surname, HouseNumber, Street, Town, Postcode)  Insurance(PolicyNumber, RegistrationNumber, DateStarted, PolicyType, ExcessAmount) |

(e)     The database is to be extended to store information about vehicle safety certificates. Each year, a vehicle must be taken to a garage where it will be tested. If the vehicle passes the test, a certificate will be issued. Each certificate will have a unique Certificate Number. Certificates will last for 12 months so the date that a certificate is issued must be recorded, as must the name of the garage that issued the certificate.

The database must keep a record of all the certificates that have been issued for each vehicle. For a particular vehicle this will include the current certificate together with any certificates that have been issued in the past.

Explain how you would change the design of the database so that the information about safety certificates can be stored.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

**(Total 18 marks)**

**Q12.**

A company sells furniture to customers of its store. The store does not keep the furniture in stock. Instead, a customer places an order at the store and the company then orders the furniture required from its suppliers. When the ordered furniture arrives at the store a member of staff telephones or e-mails the customer to inform them that it is ready for collection. Customers often order more than one type of furniture on the same order, for example a sofa and two chairs.

Details of the furniture, customers and orders are to be stored in a relational database using the following four relations:

|  |  |
| --- | --- |
|  | Furniture(FurnitureID, FurnitureName, Category, Price, SupplierName) |
|  | CustomerOrder(OrderID, CustomerID, Date) |
|  | CustomerOrderLine(OrderID, FurnitureID, Quantity) |
|  | Customer(CustomerID, CustomerName, EmailAddress, TelephoneNumber) |

(a)     These relations are in Third Normal Form (3NF).

What does this mean and why is it important that the relations in a relational database are in Third Normal Form?

Meaning: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

Why important: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(b)     On the incomplete Entity-Relationship diagram below show the degree of any **three** relationships that exist between the entities.

|  |  |  |
| --- | --- | --- |
| Furniture |  | CustomerOrder |
|  |  |  |
| Customer |  | CustomerOrderLine |

**(3)**

(c)     Complete the following Data Definition Language (DDL) statement to create the Furniture relation, including the key field.

CREATE TABLE Furniture (

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

)

**(3)**

(d)     A fault has been identified with the product that has FurnitureID number 10765. The manager needs a list of the names and telephone numbers of all of the customers who have purchased this item of furniture so that they can be contacted. This list should contain no additional details and must be presented in alphabetical order of the names of the customers.

Write an SQL query that will produce the list.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(6)**

(e)     The system requirements have changed. When an order is placed the system must now record the name of the sales person who took the order.

Place **one** tick next to the correct SQL command below that should be used to update the structure of the database so that this information can be recorded.

|  |  |
| --- | --- |
| **Command** | **Correct? (Tick one)** |
| ALTER TABLE |  |
| CREATE FIELD |  |
| INSERT COLUMN |  |

**(1)**

**(Total 17 marks)**

**Q13.**

A library uses a database management system (DBMS) to store details of the books that it stocks, its members and the loans that it has made. These details are stored in a database using the following three relations:

Book(BookID, Title, Author, Publisher)

Member(MemberID, Surname, Forename, HouseNumber, StreetName,  
Town, County, Postcode, DateOfBirth, EmailAddress)

Loan(MemberID, BookID, LoanDate, DueBackDate, Returned)

The library does not stock more than one copy of the same book.

(a)     The key in the Loan relation is made up of three attributes.

What is the name given to a key that is made up of multiple attributes?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     The relations in this database have been fully normalised.

State **two** properties that the relations in a fully normalised database must have.

Property 1: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

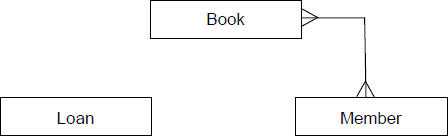
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Property 2: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     Complete the Entity-Relationship diagram below to show the degree of the **two** missing relationships between the entities.



**(2)**

(d)     The library is holding a ‘meet the author’ event at which members will be able to meet the author Lucas Bailey. The librarian wants to send e-mails to all of the library members who have read any of his books to invite them to the event.

Write an SQL query to retrieve the EmailAddress, Forename and Surname of the people to whom e-mails should be sent.

SELECT

FROM

WHERE

**(5)**

(e)     A new book is to be added to the library stock. The book details are:

•        BookID: 837023              •        Author: Karen Matu

•        Title: Kenyan Safari         •        Publisher: African Travel Guides

Write the SQL commands that will add this book into the database.

INSERT INTO \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

VALUES \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

|  |
| --- |
| The definitions of the three relations in the database at the beggining of this question are repeated here so that you can answer Question **(f)** on this page without having to turn back.  Book(BookID, Title, Author, Publisher)  Member(MemberID, Surname, Forename, HouseNumber, StreetName, Town, County, Postcode, DateOfBirth, EmailAddress)  Loan(MemberID, BookID, LoanDate, DueBackDate, Returned) |

(f)     The system requirements have changed. The library now needs to be able to stock more than one copy of the same book. Two different copies of the same book will have the same BookID.

Explain how the database design could be modified to meet this new requirement, whilst ensuring that the database remains normalised.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(3)**

(g)     The DBMS organises the data in the database in files using hashing.

(i)      Why is hashing used?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(ii)     In the context of storing data in a file, explain what a *hash function* is.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(iii)     Collisions can occur when hashing is used.

In this context, explain what a *collision* is and how one might be dealt with.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

**(Total 20 marks)**

**Q14.**

A company is building an e-commerce website. The website will display details of the products that the company sells and allow customers to place orders. Customers must register on the website before they can place an order and each order can be for one or more different products.

The product, customer and order details will be stored in a relational database. It was originally proposed that the following three relations were required:

Product(ProductNumber, ProductPrice, ProductDescription, QuantityInStock)

Order(OrderNumber, OrderDate, CustomerID, OrderingComputerIPAddress, ProductNumber, Quantity)

Customer(CustomerID, CustomerName, Address, Postcode, EmailAddress, PaymentCardNumber)

The computer programmer identified a problem with the Order relation and stated that it should be divided up into two separate relations:

Order(OrderNumber, OrderDate, CustomerID, OrderingComputerIPAddress)

OrderLine(OrderNumber, ProductNumber, Quantity)

(a)     Describe the problem that the programmer identified with the original Order relation and explain what the cause of this problem was.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(b)     Complete the Entity-Relationship diagram below to show the degree of any **three** relationships that exist between the entities.

|  |  |  |
| --- | --- | --- |
| Customer |  | Product |

|  |  |  |
| --- | --- | --- |
| Order |  | OrderLine |

**(3)**

(c)     Complete the following Data Definition Language (DDL) statement to create the Product relation, including the key field.

CREATE TABLE Product

(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(3)**

(d)     The individual web pages that describe each product will be generated dynamically using server-side scripting.

Explain what a *server-side script* is.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(e)     The definitions of the four relations in the database are repeated here so that you can answer the questions on these pages without having to turn back in the question booklet.

|  |
| --- |
| Product(ProductNumber, ProductPrice, ProductDescription, QuantityInStock)  Order(OrderNumber, OrderDate, CustomerID, OrderingComputerIPAddress)  OrderLine(OrderNumber, ProductNumber, Quantity)  Customer(CustomerID, CustomerName, Address, Postcode, EmailAddress, PaymentCardNumber) |

A customer can add a product to an order by loading the product’s web page, typing the quantity of the product required into a text box and then pressing the order button on the page. The web browser then sends the ProductNumber and Quantity to the web server.

After the user has pressed the order button, the CGI script shown below is executed.

Line No  
*1*.     ProdNum = Request("ProductNumber")  
*2*.     SaleQuant = Request("Quantity")  
*3*.     ProdDetails = ExecuteSQL("SELECT ProductPrice  
                  FROM Product WHERE ProductNumber = " + ProdNum)  
*4*.     ItemPrice = ProdDetails.GetField("ProductPrice")  
*5*.     TotalPrice = ItemPrice \* SaleQuant  
*6*.     Response.Write ("Total Price is " + TotalPrice)

(i)     Explain the purpose of lines *1.* and *2.* of the CGI script:

ProdNum = Request(“ProductNumber”)  
SaleQuant = Request(“Quantity”)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(ii)     Explain the purpose of line *3*. of the CGI script:

ProdDetails = ExecuteSQL(“SELECT ProductPrice  
         FROM Product WHERE ProductNumber = ” + ProdNum)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(iii)     Explain the purpose of line 6. of the CGI script:

Response.Write (“Total Price is ” + TotalPrice)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(f)      A web page is required that will display a summary of the products that are on a particular order.

The summary must include only the ProductNumber, ProductDescription, ProductPrice and the Quantity of the product that has been ordered. These must be displayed in ascending order of ProductNumber.

Write an SQL query that will find the data needed to produce the order summary web page for order number 4013.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

**(Total 20 marks)**

**Q15.**

A company manages subscriptions to thirty different magazines. Customers can subscribe to receive one or more of the magazines.

•        Each magazine has a category such as Gardening or Current Affairs.

•        Each magazine has a subscription rate, which is the cost of subscribing to receive the magazine for 12 months.

Details of the subscriptions are to be stored in a database using the following three relations:

Magazine(MagazineID, MagazineName, Category, SubscriptionRate)

Subscription(SubscriptionID, MagazineID, CustomerID, StartDate, EndDate)

Customer(CustomerID, CustomerName, Address, Postcode, TelephoneNumber)

(a)     These relations are in *Third Normal Form*.

What does this mean and why is it important that the relations in a database are in Third Normal Form?

Meaning: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Why important: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

(b)     Complete the Entity-Relationship diagram below to show the degree of the **three** relationships that exist between the entities.

|  |  |  |
| --- | --- | --- |
| Magazine |  | Subscription |
|  |  |  |
|  | Customer |  |

**(2)**

(c)     Complete the following Data Definition Language (DDL) statement to create the Magazine relation, including the key field.

CREATE TABLE Magazine

(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(d)     The company wants to send letters to customers with expired subscriptions to the magazine ‘AQA Computing Now’ to encourage them to subscribe again. The letters must be sent to all customers for this magazine whose subscription ended before 01/06/2010. A customer’s name, address and postcode must be included in each letter.

Write an SQL query that will find the data needed to produce the letters.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

(e)     The magazine named ‘AQA Gardening Monthlyߣ is to be renamed ‘AQA Garden Newsߣ.

Complete this SQL statement to update the data in the Magazine table to reflect this change.

UPDATE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

SET \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

WHERE \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

**(Total 15 marks)**

**Q16.**

A local free recycling group has been set up in a small community. If Member A no longer needs an item, he can advertise it on the group’s website. If Member B needs an item, she can search for it on the site. The two members then get in contact and Member A gives the item to Member B.

Items are removed from the site when exchanged or after six weeks.

The scheme is based on a relational database, which includes the two tables, **Member** and **Item**.

(a)     Complete the definition below.

A relational database is a collection of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ in which

relationships are created through \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(b)     Each table below shows a small selection of the records.

                                           Table **Member**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MemberID** | **EmailAddress** | **KnownAs** | **Password** | **Date joined** | **\_\_\_\_\_\_\_** |
| Flintstone | fred127@internetmail.abc | Fred | wonderland | 15/08/07 |  |
| Hollyberry | panbdy23@aqa.org.uk | Pam | schnuki2 | 02/09/07 |  |
| Sunnyboy | pssmith@localmail.fin | Peter | pwrrsmith | 10/10/07 |  |
| ContraryMary | mary.peters@webmail.wyz | May | thisismypassword | 15/11/07 |  |

                                           Table **Item**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ItemNumber** | **Date Posted** | **MemberID** | **ItemOnOffer** | **Description** |
| 2435 | 12/12/07 | Flintstone | Paving slabs | 15 York stone paving slabs, 600 mm square, dirty but goodcondition. |
| 2436 | 12/12/07 | Flintstone | Lawn mower | Greengrass petrol driven lawnmower. Too heavy for menow! |
| 2438 | 14/12/07 | ContraryMary | Bunk beds | Wooden bunk beds, 150 cm x 100 cm, with mattresses. |
| 2439 | 14/12/07 | ContraryMary | Chair | Rocking chair. Wooden arms. Needs new upholstery. |
| 2442 | 15/12/07 | Hollyberry | Child’s bed | Child’s bunk bed with desk area underneath. Wooden. Slightlyworn. |
| 2443 | 16/12/07 | We54rey | Chairs | 6 garden chairs, good as new. |
| 2445 | 19/12/07 | Dnukia69 | Bed | 5 ft double bed. Firm sprung mattress. |

(i)      The primary key in table **Member** is the **MemberID**. What is the purpose of a primary key?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(ii)     Why is **EmailAddress** not a good choice for use as a primary key?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(iii)     Explain, in the context of this question, the advantage of indexing the **Item** table on the field **ItemOnOffer.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(iv)    Name the foreign key in the table **Item.**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

**(Total 10 marks)**

**Q17.**

The council of a large city wants to reduce the number of cars owned by the city’s residents. The council is planning to introduce a car share club. The club will own cars, which will be parked in designated parking areas across the city when not hired out. The club members will be able to hire a car via the Internet, by e-mail or by phone. Members can book a car for one or more hours, or one or more days, up to a maximum of three weeks. Members pay a fixed monthly membership fee. They are also billed for the amount of time the car is hired plus a mileage charge. If a member returns a car late to the designated parking area, there will be a penalty charge for each additional hour.  
Each car has a built-in computer with mobile phone technology to provide a communication link to the booking centre. Each member will receive a membership card which contains an RFID (Radio Frequency Identification) tag and the member is issued with a PIN (Personal ID Number).

(a)     When a resident of the town wants to join the car share club, they are required to provide the following details:

– Credit card number

– Full Name and Address (as registered against their credit card)

– Driving Licence Number

– E-mail address

– Mobile telephone number

The monthly fee plus hire charges are automatically added to the member’s credit card.  
A statement of charges is available for the member to download from the Internet.

(i)      What other details are required to be stored about the member so that the member can only access their own statement of charges?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

The details held about each Parking Area are:

         – A 3-character unique Location Code

         – Name of Parking Area, such as “Station”

         – Post Code of Parking Area

The details held about each car are:

         – Car Registration Number

         – Designated Parking Area

At the time of booking, the member will be asked to provide the following details:

         – Member ID

         – Pick-up point (from which parking area)

         – Hire start date and time

         – Hire end date and time

The booking centre allocates a car from the chosen parking area to the member and issues the member with a booking reference code.

Choosing suitable attribute identifiers, complete the relations making sure that the primary key attribute(s) are underlined.

(ii)     Member ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(1)**

(iii)     ParkingArea ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                        \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(1)**

(iv)    Car ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

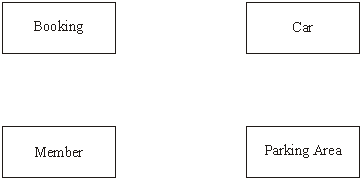
         \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(1)**

(v)     Booking ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

(b)     Complete the entity-relationship diagram for the entities below:



**(3)**

(c)     Using SQL commands SELECT, FROM, WHERE, ORDER BY, write an SQL statement to query the database tables to produce a list of bookings for the month of December 2007. The results of the query are to be in member ID order.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

**(Total 15 marks)**

**Q18.**

Give **one** data definition language (DDL) command that could be found in a script used to construct a relational database table.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(Total 1 mark)**

**Q20.**

A chef keeps her recipes on a single-table database system**. Figure 1** shows the Recipe Table.

**Figure 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Recipe ID** | **Dish** | **Recipe Ingredients (including quantity)** | **Preparation Time** | **Cooking Time** | **Number of Servings** | **Cooking Instructions** |
| 1 | Hummus | 250g chickpeas 6 cloves garlic 50ml lemon juice 340g tahini | 20 minutes | 2 hours | 8 | Cook chickpeas until soft. Puree in food processor. Add remainder of ingredients, mix well. |
| 2 | Feta Salad | 400g tomatoes 250g feta cheese 1 cucumber 50g olives 45ml vinaigrette | 15 minutes | none | 4 | Mix all salad ingredients together. Season with salt and pepper. Dress with vinaigrette. |
| 3 | Casserole | 500g chickpeas 400g tomatoes 450g potatoes | 10 minutes | 2 hours | 4 | Cook chickpeas until nearly soft. Add cubed potatoes and tomatoes........ |
| : | : | : | : | : | : | : |
| : | : | : | : | : | : | : |

The chef’s only supplier provides her with an on-line price list for her ingredients. **Figure 2** shows the PriceList Table.

**Figure 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **FoodItemID** | **FoodItemName** | **PackSize** | **Price** |
| Tom001 | Tomatoes | 400g | £0.55 |
| Chi002 | Chickpeas | 250g | £0.75 |
| Cuc003 | Cucumber | single | £0.50 |
| : | : | : | : |

(a)     (i)      Which of the above two tables is **not** in First Normal Form? \_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(ii)     Why? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     The chef would like to be able to cost the ingredients for her dishes and is asking you to help her design a database which would allow the costing of dishes without retyping any of the pricelist details.

Database theory states that database tables should be *fully normalised*

(i)      What does fully normalised mean?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(ii)     Why is it desirable that tables are fully normalised?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(c)     When the data in **Figure 1** and **Figure 2** are stored in a fully normalised relational database, three relations, **Recipe, FoodItem** and **RecipeIngredient** are needed. For each of these, complete the relations, making sure the primary key attribute(s) are underlined.

(i)      Recipe( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(1)**

(ii)     FoodItem ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(1)**

(iii)     RecipeIngredient( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(4)**

(d)     Using the SQL commands SELECT, FROM, WHERE, ORDER BY and any others that you consider appropriate, write the SQL statement to list all the ingredients and their quantities required for Feta Salad. The results table should also show the pack size of the food item and the associated price. The list should be in alphabetical order of ingredient.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

**(Total 16 marks)**

**Q21.**

The following are parts of three tables in a relational database for a book loan system.

Member

|  |  |  |  |
| --- | --- | --- | --- |
| MemberID | Surname | Forename | TelephoneNumber |
| **...** | **...** | **...** | **...** |
| **IV270023** | **Smith** | **Gerald** | **01234 567890** |
| **IV270024** | **Smith** | **Wendy** | **01234 567890** |
| **IV280016** | **Pope** | **Anne** | **01234 465987** |
| **IV280017** | **Patel** | **Arwen** | **01234 657980** |
| **...** | **...** |  | **...** |

Book

|  |  |  |  |
| --- | --- | --- | --- |
| BookID | Title | Author | Value |
| **...** | **...** | **...** | **...** |
| **1457X** | **Travels with my Family** | **A M MacIntyre** | **£13.50** |
| **14582** | **Travels with my Family** | **AM MacIntyre** | **£13.50** |
| **15635** | **By Bicycle to Bangor** | **A M MacIntyre** | **£14.75** |
| **16370** | **Walking in Wonderland** | **BG O’Connor** | **£15.99** |
| **...** | **...** | **...** | **...** |

Loan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BookID | MemberID | DateOut | DateIn | DateReturned |
| **...** | **...** | **...** | **...** | **...** |
| **1457X** | **IV270023** | **07/12/06** | **28/12/06** | **22/12/06** |
| **16370** | **IV270024** | **07/12/06** | **28/12/06** | **22/12/06** |
| **15635** | **IV270024** | **07/12/06** | **28/12/06** | **22/12/06** |
| **...** | **...** |  | **...** | **...** |

(a)     How are relationships between entities implemented with relational database software?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     Give the primary key for the entity Loan.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     The Book table has an index on the attribute Author. Why is indexing used?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(d)     The last digit of the BookID is used for validation. What type of validation control is this an example of?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

**(Total 5 marks)**

**Q22.**

The network manager of a college has to ensure that all software on college computers is installed legally.

(a)     Which law is the network manager following?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

Some software is bought with a site licence for use on any number of college computers whereas other software is bought with a multi-user licence for installation on a specified number of college computers. The network manager wants to set up a relational database to keep details of software licences and which college computers have which software installed. She has identified the following details she wants to store about each software package:

•        Software name

•        Version

•        Software ID (ten alphanumeric characters), unique

•        Supplier

•        Date purchased

•        Expiry date

•        Number of computers licenced

When any software is installed on a college computer, she wants to store the following details:

•        Computer ID of the computer on which the software is to be installed (six alphanumeric characters)

•        Date of software installation

•        Staff ID who requested the software (3 letters) for this computer

(b)     The two entities **SoftwareLicence** and **SoftwareInstallation** have been identified.

Draw an Entity-Relationship diagram for the above entities.

**(3)**

(c)     Complete the following Data Definition Language (DDL) statements to create the fully normalised tables necessary to implement the database, including all key fields.

(i)      CREATE TABLE SoftwareLicence

( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(3)**

(ii)     CREATE TABLE SoftwareInstallation

( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(4)**

(d)     The network manager wants a list of each computer and what software is installed on it. Using the SQL commands SELECT, FROM, WHERE, ORDER BY and any others that you consider appropriate, write an SQL statement to query the database tables for a list of computers with software name and version installed.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

**(Total 15 marks)**

**Q23.**

Sam starts work at a small business selling garden tools and equipment by mail order.

Four of the employees in the business have their own stand-alone computers.

•        Mary takes the order and enters it into her computer.  She records the customer’s name and address, plus the item(s) ordered. She prints out a copy and gives it to Renie.

•        Renie takes the items off the shelves and packs the order. She records, on her computer, the customer’s name and address, and the items despatched. She notes any item that is out of stock; these will have to follow in a later despatch. She prints off a copy of these details to give to Hassan.

•        Hassan produces the invoice. He records, on his computer, the customer’s name and address and the items despatched with prices. He adds a handling charge and puts the invoice into an envelope ready for posting to the customer. He gives a copy invoice to James.

•        James is the accountant. He records, on his computer, the customer’s name and address, the invoice amount and whether it has been paid or not.

(a)     This system has disadvantages. Explain **two** of these.

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(4)**

(b)     Sam suggests that if the computers were networked, they could share files. If the computers were networked, why is it unlikely that they could share files as they are set up currently?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(c)     The company decide to start again and create a relational database. Sam starts talking about *attributes, primary keys* and *foreign keys.*

(i)      Define an attribute.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(ii)     Define a primary key.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(iii)     Define a foreign key.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(d)     Initially Sam thinks that four tables are needed

1.      Customer table

2.      Stock table containing details of each item of stock

3.      Order table containing general details of each order

4.      OrderLine table containing details of each item ordered for a particular order

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Table: | Customer |  | Stock |  | Order |  | OrderLine |
| To include these fields | Surname  Firstname  Address 1  Town  PostCode  Telephone Number  CustomerID |  | PartNumber  Description  UnitPrice  NumberInStock |  | OrderNumber  DateOfOrder  CustomerID |  | OrderNumber  OrderLineNumber  PartNumber  Quantity |

(i)      What would be the most suitable primary key for the table Customer?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(ii)     What would be the most suitable primary key for the table OrderLine?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(iii)     Name a foreign key in the table OrderLine.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

**(Total 12 marks)**

**Q24.**

A computer technician in a school is asked to keep up-to-date details of the hardware equipment the school owns (known as an inventory). The school management require the following details to be stored for each item of hardware:

•        description

•        make

•        model

•        inventory reference number (20 alphanumeric characters)

•        date of purchase

•        purchase price

•        room where item is kept.

If an item is loaned, the following details must be stored:

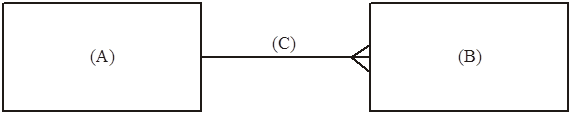
•        location details of where the item will be located

•        the initials of the person responsible for its return

•        the dates of removal and return.

(a)     The **two** entities **HardwareItem** and **EquipmentLoan** have been identified.

(i)      Complete the diagram below.



A \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

B \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

C \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(2)**

(ii)     Name this type of diagram.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     Complete the following Data Definition Language (DDL) statements to create the fully normalised tables necessary to implement the database.

CREATE TABLE HardwareItem

( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ )

CREATE TABLE EquipmentLoan

( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ )

**(6)**

(c)     Using the SQL commands SELECT, FROM, WHERE, ORDER BY and any others that you consider appropriate, write an SQL statement to query the database tables for the description, inventory reference number and date of removal of those items of equipment that have been loaned since a given date. The results of this query should be displayed in such a way as to make it easy to see how many times each item was loaned.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(6)**

**(Total 15 marks)**

**Q25.**

A head of department in a school wishes to store data on her pupils’ module results for AS and A-level Computing. The data requirements are defined as follows:

•        each pupil has their forename(s) and surname recorded;

•        each pupil is assigned a unique candidate number;

•        each module is identified by a module code;

•        each module has a given maximum number of marks available;

•        each module is available each year at a winter and/or summer session;

•        each pupil’s module result is a number of marks between 0 and the maximum for that module;

•        each pupil may resit a module several times, the best results being used to calculate the overall grade.

A single table, ResultsTable, was constructed initially in a relational database. **Figure 1** shows the structure of this table and a few entries.

**Figure 1**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Pupil Forenames | Pupil Surname | Candidate Number | Module Code | Exam Session | Module Mark | Level | Total Mark | Grade |
| Ali | Patel | 1234 | CPT1 | W04 | 54 | AS | 187 | C |
|  |  |  | CPT1 | S04 | 74 | A | 318 | D |
|  |  |  | CPT2 | S04 | 63 |  |  |  |
|  |  |  | CPT3 | S04 | 50 |  |  |  |
|  |  |  | CPT4 | W05 | 43 |  |  |  |
|  |  |  | CPT2 | S05 | 60 |  |  |  |
|  |  |  | CPT5 | S05 | 43 |  |  |  |
|  |  |  | CPT6 | S05 | 45 |  |  |  |
| Marie | Frost | 1357 | CPT1 | W04 | 97 | AS | 255 | A |
| Anne |  |  | CPT2 | S04 | 86 | A | 500 | A |
|  |  |  | CPT3 | S04 | 72 |  |  |  |
|  |  |  | CPT4 | W05 | 50 |  |  |  |
|  |  |  | CPT4 | S05 | 72 |  |  |  |
|  |  |  | CPT5 | S05 | 75 |  |  |  |
|  |  |  | CPT6 | S05 | 98 |  |  |  |
| John Mark | Smith | 2345 | CPT1 | W05 | 65 | AS | 169 | D |
|  |  |  | CPT1 | S05 | 60 |  |  |  |
|  |  |  | CPT2 | S05 | 72 |  |  |  |
|  |  |  | CPT3 | S05 | 32 |  |  |  |
| Ali : | Patel : | 7315 : | : : | : : | : : | : : | : : | : : |

(a)     Which of the column heading(s) in ResultsTable would be suitable as a primary key?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     What makes this table **not** in First Normal Form?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(c)     When the data in Figure 1 is stored in a fully-normalised relational database, **three** relations, **Pupil**, **ModuleResult** and **PupilGrade** are used. For each of these, complete the relations making sure the primary key attribute(s) are underlined.

(i)      Pupil (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(ii)     ModuleResult (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

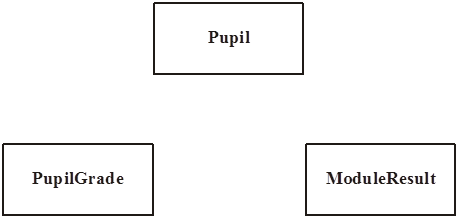
(iii)     PupilGrade (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(d)     Complete the partial Entity-Relationship diagram in **Figure 2** to show the degree of **two** relationships which exist between the given entities.

**Figure 2**

****

**(2)**

(e)     Using the SQL commands SELECT, FROM, WHERE, ORDER BY and any others that you consider appropriate, write an SQL statement to query the database tables for the pupil forenames, surname and A-level grades in descending order of total mark.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

**(Total 15 marks)**

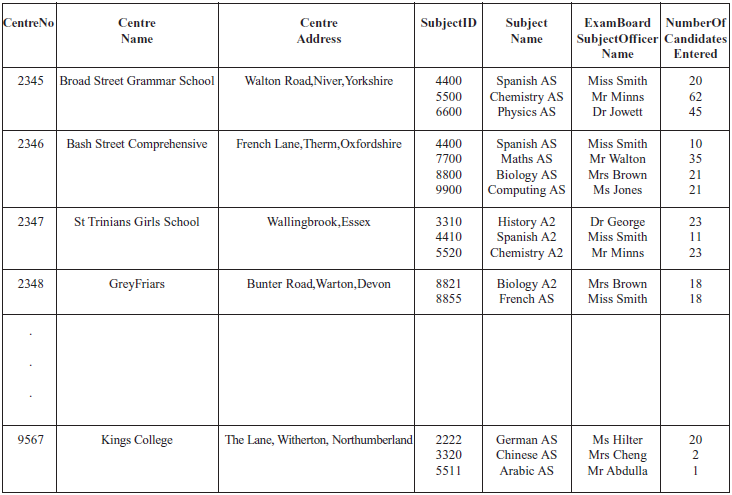
**Q26.**

An Examination Board records total entries by centre and subject for its examinations for June 2005 in an un-normalised relational database table as shown in the figure below. The data requirements specify that

•        a subject offered for examination by the Examination Board has at most one Examination Board Subject Officer;

•        an Examination Board Subject Officer may be a subject officer for more than one subject offered for examination by the Examination Board;

•        Centre Number and SubjectID are unique.



The relation for this table is as follows

**ExamBoardEntryNumbers**(CentreNo. CentreName, CentreAddress, SubjectID, SubjectName, ExamBoardSubjectOfficerName, NumberOfCandidatesEntered)

(a)     What makes this table un-normalised?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(1)**

(b)     When the data in the figure above is stored in a fully-normalised relational database three relations Centre, **CentreEntryNumber and Subject are used. For each of these** complete the relations making sure that the primary key attribute(s) are underlined.

(i)      Centre ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(ii)     CentreEntryNumber (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(iii)     Subject (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**(2)**

(c)     (i)      Complete the entity-relationship diagram for the entities **Centre** and **CentreEntryNumber.**

|  |  |  |
| --- | --- | --- |
| Centre |  | CentreEntryNumber |

**(1)**

(ii)     Complete the entity-relationship diagram for the entities **Centre** and **Subject**.

|  |  |  |
| --- | --- | --- |
| Centre |  | Subject |

**(1)**

(d)     Problems with particular entries are also logged in the relational database. These problems are reported in e-mails sent to the Examination Board by the Examination Officer responsible for a centre’s examination entries. The Examination Board may communicate a response via e-mail. The relational database includes two extra fully-normalised relations **ExaminationOfficer** and **Problem** for this purpose.

**ExaminationOfficer**(CentreNo, ExamOfficerSurname, Title, EMailAddress)

**Problem**(ProblemId, DateReported, ProblemDescription, CentreNo, ReplySent)

Using the SQL commands

SELECT, FROM, WHERE, ORDER BY

and any other commands which are considered appropriate, write an SQL statement to query the database tables for the surname and centre number of all Examination Officers who have reported a problem before 1st March 2005 and the corresponding description of the problem.

The result of the query is to be ordered in ascending order of CentreNo.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**(5)**

(e)     Name the type of package which would be most suitable to use with the database or on its own for creating a mail merge operation to send personalised letter attachments with each e-mail.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

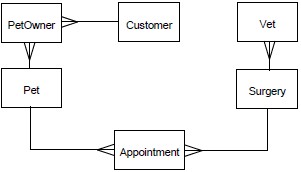
**(1)**

**(Total 15 marks)**

Mark schemes

**Q1.**

(a)  **All marks AO2 (analysis)**

****

**1 mark**: Entity Appointment added and correct relationships and degree drawn to Pet and Surgery entities.

**1 mark:** Entity Customer added and correct relationship and degree drawn to PetOwner. **Note**: If PetOwner relation not created then allow this mark if a many-to-many relationship is drawn between Pet and Customer, even though this is not fully normalised.

**1 mark**: Entity PetOwner added and correct relationship and degree drawn to Pet.

**A.** entity names do not have to match diagrams exactly but must convey same purpose.

**Ignore the inclusion of any additional entities and the drawing of any other relationships, whether they are correct or not.**

**3**

(b)  **All marks AO2 (analysis)**

Customer(CustomerID, Forename, Surname, TelephoneNumber)

Appointment(PetID, Date, Time, SurgeryName)

PetOwner(CustomerID, PetID)

**1 mark**: Customer relation created and contains the correct attributes and CustomerID identified as the entity identifier.

**I.** any additional reasonable attributes, including PetID if the PetOwner relation has not been created, **but** reject PetID included if the PetOwner entity has been created. If PetID is included then it is acceptable for it to be in or not in the entity identifier.

**1 mark**: Appointment relation created and contains the correct attributes.

**I.** any additional reasonable attributes including CustomerID, VetID

**A.** date and Time given as one combined field.

**1 mark**: Composite entity identifier of PetID, Date and Time identified for Appointment relation.

**A.** creation of a new attribute to be the entity identifier eg AppointmentID.

**A.** date and Time given as one combined field.

**1 mark**: PetOwner relation created and contains the correct attributes and no others. Additionally, either:

•   both attributes identified as a composite entity identifier, or,

•   **A.** a new entity identifier eg OwnershipID created.

**R.** Just one of PetID or CustomerID given as entity identifier.

**For all mark points**

**A.** alternative names for relations and attributes created by candidate, so long as meaning is clear.

**R.** use of incorrect attribute names for attributes already named in question paper.

**A.** spaces in relation and attribute names.

**I.** if any unnecessary relations are created.

**I.** any representation for foreign keys.

**Accept responses written in SQL – ignore syntactical errors and data type errors in such responses.**

**4**

(c)  **All marks AO2 (analysis)**

Torquay / the town name is missing quotations marks // needs quotation marks around it;

There is no linking condition/join between the two tables using the SurgeryName // a join needs to be added to the FROM clause using the SurgeryName // a linking condition needs to be added to the WHERE clause using the SurgeryName // the condition Surgery.SurgeryName = Vet.SurgeryName is missing / needs adding;

There is no linking condition/join between the two tables using the SurgeryName // a join needs to be added to the FROM clause using the SurgeryName // a linking condition needs to be added to the WHERE clause using the SurgeryName // the condition Surgery.SurgeryName = Vet.SurgeryName is missing / needs adding;

**NE.** the tables have not been linked.

**2**

(d)  **All marks AO1 (understanding)**

Two users (read and) edit a record/the same data simultaneously;

**NE.** access/read unless later made clear the record/data is changed

**NE.** edit the database simultaneously

One user writes the record/data back/saves then the other user writes the record/data back/saves;

One user’s update is lost // only one user’s update is the kept; **NE.** data is lost

**A.** examples that map to the above points.

**If no other marks awarded, award one mark for the use of the term “lost update problem”.**

**Refer examples relating to data being read whilst another transaction that is later rolled-back is in progress to team leaders.**

**3**

**[12]**

**Q2.**

**All marks AO1 (knowledge)**

**Mark against “Record locks” or “Timestamp ordering” mark scheme, depending upon which method the student has selected.**

**Record locks**:

When a transaction on a record starts / when a user starts to edit a record an (exclusive) lock is set on the record; **R.** database/data/file/table for record

Other transactions/users cannot edit (**A.** access) the record/data until the lock is released/while the lock is in place/until the first edit is completed;

**Timestamp ordering:**

Timestamps are generated for each transaction // timestamps indicate the order that transactions occurred in; **A.** timestamps generated for edits/queries as BOD

Database records timestamp of last read / last write transaction for each record / data item; **A.** just one of read/write

Database server applies rules to determine if processing a transaction will result in loss of data integrity/inconsistency (and if so aborts the transaction); **A.** Examples of rules for this mark point:

•   If a transaction tries to write to a record/data item then the transaction should be aborted if the read/write timestamp on the record/data item is greater that the time at which the transaction started.

•   If a transaction tries to read a record/data item then the transaction should be aborted if the write timestamp on the record/data item is greater that the time at which the transaction started.

**Max 2**

**[2]**

**Q3.**

**All marks AO1 (understanding)**

|  |  |  |
| --- | --- | --- |
| **Level** | **Description** | **Mark Range** |
| 4 | A line of reasoning has been followed to produce a coherent, relevant, substantiated and logically structured response. The response covers all three areas indicated in the guidance below and in at least two of these areas there is sufficient detail to show that the student has a good level of understanding. To reach the top of this mark range, a good level of understanding must be shown of all three areas. | 10-12 |
| 3 | A line of reasoning has been followed to produce a coherent, relevant, substantiated and logically structured response which shows a good level of understanding of at least two areas indicated in the guidance below. | 7-9 |
| 2 | A limited attempt has been made to follow a line of reasoning and the response has a mostly logical structure. At least four points have been made. Either a good level of understanding of one area from the guidance has been shown or a limited understanding of two areas. | 4-6 |
| 1 | A few relevant points have been made but there is no evidence that a line of reasoning has been followed. The points may only relate to one or two of the areas from the guidance or may be made in a superficial way with little substantiation. | 1-3 |

**Guidance – Indicative Response**

**For each guidance point, if the student expands on the point to explain in what way the measure will improve performance then this can be considered to be a second point.** For example:

•   “Using a processor with more cores” is one point.

•   “Using a processor with more cores which will be able to execute multiple instructions simultaneously” is two points.

Note that just “faster” is not enough to count as an expansion point without an explanation of why.

**1. Server Hardware**

Replace the processor with one which has more cores

Replace the processor with one which has more cache memory // increase the amount of cache memory

Replace the processor with one which runs at a faster clock speed **NE.** faster processor

Use a parallel processor architecture // use more processors which can work in parallel

Use a processor with a bigger word size

Use a processor that makes (better) use of pipelining

Install more RAM // main memory // primary memory

Use RAM // main memory // primary memory with a faster access time

Replace HDDs with SSDs // Replace HDDS with HDDs that can read data at a faster rate

Defragment the HDD

Replace the motherboard with one which has buses which run at a faster clock speed

Replace the motherboard with one which has more lines in the data bus

Use the Harvard architecture

Distribute the processing across multiple servers

**2. Network**

Replace the network cable with cable that has a higher bandwidth // replace copper cable with fibre-optic cable **A.** Ethernet cable for fibre-optic NE. higher bandwidth network

Replace any wireless / WiFi connections with wired ones

Replace the network cards with ones that can transmit data at a higher bitrate

Consider the overall network design eg how the network is divided into subnets **A.** split the network into subnets

Use a star topology (instead of a bus)

Consider using a more efficient protocol for the data across the network

Add additional wireless access points

**3. Database and Software**

Use a more efficient technique for controlling concurrent access to the database // replace record/table locks with serialisation/timestamp ordering/commitment ordering

Replace the database software with software that uses more efficient algorithms for tasks **A.** examples eg replace linear search with binary search

Use the index feature of the database to speed up searching on fields that are commonly used for this purpose

Rewrite the database software in a language that is suitable for concurrent execution // use a functional programming language for the database software

Ensure the software is compiled rather than executed by an interpreter // rewrite the software in assembly language/machine code

Review the conceptual model of the database to see if it contains any inefficiencies such as data redundancy that could be eliminated **A**. normalise the database design

Consider if it would be appropriate to sacrifice normalisation of the conceptual model to improve performance

Use a non-relational database system **A.** examples eg NoSQL

Distribute the data across multiple servers

Try to reduce the amount of other (unrelated) software that might be running on the database server at the same time

Try to reduce the number of database accesses that need to be made simultaneously // run some tasks at quiet times / overnight

Purge / archive data that is no longer necessary / in use

**[12]**

**Q4.**

(a)  **All marks AO2 (analyse)**

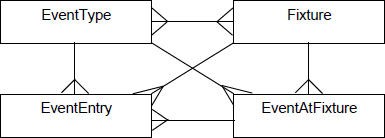
**1 mark** for any one correctly drawn relationship

**OR**

**2 marks** for three relationships drawn correctly

**Max 1** if more than three relationships drawn and any are incorrect

**A.** a many:many relationship drawn between EventType and Fixture as this is modelled by a linking relation (EventAtFixture)



**2**

(b)  **All marks AO2 (analyse)**

There is no data type for the primary key / AthleteID // The primary key / AthleteID needs a data type;

The data type is specified before the fieldname // fieldname should precede the data type // PRIMARY KEY is specified before the fieldname; **A.** an example of a specific field and data type which are the wrong way around

There is a semi-colon missing at the end;

**Max 2**

**2**

(c)  **All marks AO1 (understanding)**

\*Minimise data duplication // no unnecessary repeated data; **A.** reduce for minimise **R.** eliminate

\*Eliminate data redundancy; **A.** reduce/minimise for eliminate

Eliminate data inconsistency // improve consistency // avoid inconsistency problems;

Eliminate update anomalies; **A.** example in context **A.** updates only need to be made in one place

Eliminate insertion anomalies; **A.** example in context

Eliminate deletion anomalies; **A.** example in context

**NE.** easier to update/insert/delete without concrete example or good explanation

**NE.** fewer errors when updating / inserting / deleting without concrete example or good explanation

**NE.** saving space / memory

**NE.** easier / faster to query

**Note: Only award one of the two marks with \*. ie a response cannot get two marks for discussion of only duplication and redundancy**

**2**

(d)  **3 marks for AO2 (analyse) and 2 marks for AO3 (programming)**

Mark Scheme

**AO2 (analyse) – 3 marks:**

**1 mark** for correctly analysing the data model and identifying the tables that data needs to be extracted from (Athlete, EventEntry, Fixture) and the fields that need to be extracted (Surname, Forename, DateOfBirth), and including these and no other tables or fields in the query

**1 mark** for correctly identifying how the data in the required tables should be combined to produce the desired result (the linking conditions - Athlete.AthleteID = EventEntry.AthleteID and EventEntry.FixtureID = Fixture.FixtureID)

**1 mark** for identifying the correct condition to use within the model for the FixtureDate field (FixtureDate = "17/09/2018") and for using the correct logical operators between all of the conditions (if a linking condition is also used)

**Note:** The AO2 marks for analysing the data model should be awarded regardless of whether correct SQL syntax is used or not as they are for data modelling, not syntactically correct SQL programming

**AO3 (programming) – 2 marks:**

**1 mark** for fully correct SQL in two of the four clauses (SELECT, FROM, WHERE, ORDER BY)

**OR**

**2 marks** for fully correct SQL in all four clauses (SELECT, FROM, WHERE, ORDER BY)

Note: For an SQL clause to be counted as “fully correct”, the syntax of the clause must be correct and the relevant AO2 decisions must also have been taken for the clause. eg the SELECT clause must have the correct fields in it only

Example Solutions

**Example 1**

SELECT Surname, Forename, DateOfBirth

FROM Athlete, EventEntry, Fixture

WHERE FixtureDate = "17/09/2018"

  AND Athlete.AthleteID = EventEntry.AthleteID

  AND EventEntry.FixtureID = Fixture.FixtureID

ORDER BY Surname

**Example 2**

SELECT Surname, Forename, DateOfBirth

FROM Athlete INNER JOIN EventEntry ON Athlete.AthleteID =

  EventEntry.AthleteID INNER JOIN Fixture ON

  EventEntry.FixtureID = Fixture.FixtureID

WHERE FixtureDate = "17/09/2018"

ORDER BY Surname

**Overall Max 4 if solution does not work fully**

Additional Guidance

**AO2 marks:**

Mark(s) can be awarded for the correct logical conditions even if the required tables are not identified as being used by the query

Allow the inclusion of the unnecessary table EventAtFixture for AO2 and AO3 marks but only if it is linked to the other tables with a correct condition i.e.

EventAtFixture.FixtureID = Fixture.FixtureID or alternatively EventAtFixture.FixtureID = EventEntry.FixtureID or both

Allow omission of delimiters around date for AO2 marks only.

**AO3 marks:**

**A.** table names before fieldnames separated by a full stop

**A.** use of Alias / AS command e.g. FROM Athlete AS A then use of A as the table name but note that command Alias is not required e.g. FROM Athlete A

**A.** INNER JOIN written as one word i.e. INNERJOIN

**A.** ORDER BY written as one word i.e. ORDERBY

**A.** ASC at end of ORDER BY clause but **R.** ASCENDING

**A.** insertion of spaces into fieldnames

**A.** use of " # or ' as delimiters around date – **Note**: delimiters are required for AO3 correct code but not for AO2 mark for date condition

**A.** date parts given in any order so long as they are separated by /

**A.** 18 instead of 2018 in year

**I.** unnecessary brackets

**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause

**DPT** for fieldname before table name

**Refer responses using nested SQL queries to team leaders.**

**5**

**[11]**

**Q5.**

(a)  **Mark is for AO2 (analyse)**

CarRegNo and JobDate;

**A**. Just both these attribute names written with no further explanation

**R**. “CarRegNo or JobDate”

**1**

(b)  **1 mark for AO2 (analyse) and 1 mark for AO1 (understanding)**

**AO2 (analyse) – 1 mark:**

A person may own more than one car // a person may bring different cars to the garage;

It might be desired to store details of an owner when the car they own is not yet known;

**A**. A car might be owned by more than one person (at different times)

**A**. Easier to transfer car from one owner to another

**AO1 (understanding) – 1 mark:**

Avoid storing owner details once for each car they own / multiple times;

Avoid having to input owner details once for each car they own;

To transfer car between owners would only have to change one attribute in the car relation;

Minimise data duplication // no unnecessary repeated data; **A**. Reduce for minimise

Eliminate data redundancy; **A**. Reduce/minimise for eliminate

Eliminate data inconsistency // improve consistency // avoid inconsistency problems;

Eliminate update anomalies; **A**. Example in context

Eliminate insertion anomalies; **A**. Example in context

**NE**. Fewer errors when updating/inserting/deleting without concrete example or good explanation

**NE**. Saving space/memory

**NE**. Easier to query

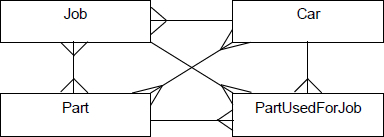
**2**

(c)  **All marks AO2 (analyse)**

**1 mark** for any one correctly drawn relationship **OR**

**2 marks** for three relationships drawn correctly

**MAX 1** if more than three relationships drawn and any are incorrect



**2**

(d)  **1 mark for AO2 (analyse) and 2 marks for AO3 (programming)**

Mark Scheme

**AO2 (analyse) – 1 mark:**

**1 mark** for correctly identifying the table in the data model that needs to be updated (Job) and the condition that should be used to identify the correct record in the table to update (JobID = 206).

**Note:** The AO2 mark for analysing the data model should be awarded regardless of whether correct SQL syntax is used or not as it is for data modelling, not syntactically correct SQL programming

**AO3 (programming) – 2 marks:**

**1 mark** for correct SQL syntax in two of the three clauses (UPDATE, SET, WHERE) **OR**

**2 marks** for fully correct SQL

Example Solution

UPDATE Job

SET JobDuration = "01:30"

WHERE JobID = 206

Additional Guidance

**AO3 marks:**

**A** Any type of quotation marks or hashes for delimiters for JobDuration or no delimiters

**A**. The value 206 if it is delimited by any type of quotation mark

**A**. Any sensible format for the time data eg "01.30", "1:30", "1:30.00" etc.

**A**. Time given as a decimal ie 1.5

**A**. Table name given before fieldname

**I**. Quotation marks around fieldnames

**I**. Any attempt to also change value of InGarage

**3**

(e)  **All marks AO3 (programming)**

**Method 1:**

INSERT INTO PartUsedForJob

VALUES (206,12,2)

**Method 1:**

INSERT INTO PartUsedForJob (JobID, PartID, QuantityUsed)

VALUES (206,12,2)

**1 mark** for correct INSERT INTO clause

**1 mark** for correct VALUES clause

**MAX 1** if SQL not fully working eg because of extra clauses

**A**. List of fields in any order for method 2, but to get the VALUES mark in method 2, order of fields list in INSERT INTO must match order of values in VALUES

**A**. The value(s) 206 and 12 if they are delimited by any type of quotation mark

**2**

(e)  **3 marks for AO2 (analyse) and 2 marks for AO3 (programming)**

Mark Scheme

**AO2 (analyse) – 3 marks:**

**1 mark** for correctly analysing the data model and identifying the tables that data needs to be extracted from (Part, PartUsedForJob) and the fields that need to be extracted (PartID, Description, Price, QuantityUsed), and including these and no other tables or fields in the query **A**. Including the table Job which is not needed, as long as it is correctly linked in by a condition

**1 mark** for correctly identifying how the data in the required tables should be combined to produce the desired result (the linking condition - PartUsedForJob.PartID = Part.PartID)

**1 mark** for identifying the correct conditions to use within the model for the JobID field (JobID = 93) and for using the correct logical operators between all of the conditions (if a linking condition is also used)

**Note:** The AO2 marks for analysing the data model should be awarded regardless of whether correct SQL syntax is used or not as they are for data modelling, not syntactically correct SQL programming

**AO3 (programming) – 2 marks:**

**1 mark** for correct SQL in two or three of the four clauses (SELECT, FROM, WHERE, ORDER BY) **OR**

**2 marks** for fully correct SQL

Example Solutions

**Example 1**

SELECT PartID, Description, Price, QuantityUsed

FROM Part, PartUsedForJob

WHERE JobID = 93

  AND PartUsedForJob.PartID = Part.PartID

ORDER BY PartID

**Example 2**

SELECT PartID, Description, Price, QuantityUsed

FROM Part INNER JOIN PartUsedForJob ON

PartUsedForJob.PartID = Part.PartID

WHERE JobID = 93

ORDER BY PartID

**Overall MAX 4 if solution does not work fully**

Additional Guidance

**AO2 marks:**

Mark(s) can be awarded for the correct logical conditions even if the required tables are not identified as being used by the query

Ignore unnecessary clause PartUsedForJob.JobID = Job.JobID

**AO3 marks:**

Accept table names before fieldnames separated by a full stop.

Accept use of Alias/AS command eg FROM Part AS P then use of P as the table

name but note that command Alias is not required eg FROM Part P.

Accept INNER JOIN written as one word ie INNERJOIN or just as JOIN

Accept ORDER BY written as one word ie ORDERBY.

Accept ASC at end of ORDER BY clause.

Accept insertion of spaces into fieldnames.

Accept use of " or ' as delimiters around number 93.

Ignore unnecessary brackets.

**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.

**DPT** for fieldname before table name.

For the **DPT** points, the penalisation is in terms of number of clauses of SQL code not marks ie if fieldname is before table name in two out of four clauses of SQL then this could count as three clauses of correct SQL

**Refer responses using nested SQL queries to team leaders.**

**Refer responses using RIGHT JOIN OR LEFT JOINT to team leaders.**

**5**

(f)  **All marks AO2 (analyse)**

**1 mark:** Create a new relation to identify which make/model(s) of car each part can be fitted to;

**A**. Use of a relation name that clearly identifies the purpose eg PartToFitMakeModel instead of an explanation

**A**. If it is just stated that a new relation is creation if the attributes in the relation make its purpose clear

**NE**. A relation to link the Part and Car relations

**2 marks from:**

Store the attributes PartID, Make and Model in the new relation;

**I**. Inclusion of additional attributes

Make the PartID, Make and Model / all the attributes the entity identifier;

**A**. The creation of a new field as an entity identifier for this relation if it is explained that a constraint would also need to be added to ensure that it is not possible to record twice in the relation that a particular part could be fitted to a particular make and model of car

Accept answers by example, such as: PartToFitMakeModel(PartID, Make, Model)

**Alternative Response**

**1 mark:**

Create two new relations, one to associate an entity identifier with each make and model of car (eg MakeModelID) and one to link the parts to this new relation

**A.** If it is just stated that new relations will be created if the attributes in the relations make their purpose clear

**2 marks from:**

Store the attributes Make and Model with a new entity identifier (eg MakeModelID) in one of the new relations;

Store the PartID in the other new relation together with the entity identifier from the first new relation (eg MakeModelID);

Make the PartID and MakeModelID the entity identified in the second new relation;

**A**. The creation of a new field as an entity identifier for this relation if it is explained that a constraint would also need to be added to ensure that it is not possible to record twice in the relation that a particular part could be fitted to a particular make and model of car

Accept answers by example, such as: UniqueMakeModel(MakeModelID, Make, Model) and PartToFitMakeModel(PartID, MakeModelID)

**A**. Table or entity for relation.

**A**. Field for attribute.

**A**. Primary key for Entity Identifier.

**3**

**[18]**

**Q6.**

(a)  Only one (type of) product per order // Must make new order for each (type of) product;

as ProductNumber / product details stored in relation that has OrderNumber as primary key / product relation directly related to order relation // there is transitive/(**A**.non-key) dependency // as relations not (fully) normalised;

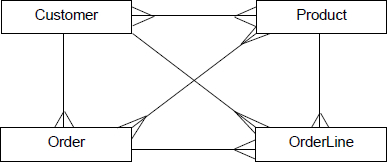
Difficult to query // requires (unnecessarily) complex queries;

as contains repeating groups (of attributes); **A**. Either way round

**A**. Table for relation

**Max 2**

(b)



**1 mark** for each correct relationship, up to **MAX 3**

**MAX 2** if more than three relationships drawn and any of them are incorrect

**3**

(c)  Sequence of instructions / program / code; **NE**. Programming language **Note**: Do not award mark for program if candidate clearly means HTML

which is executed/run/interpreted on the server (instead of the client);

executed/run/interpreted when a web page is requested;

to generate a web page (and its contents) which the server returns to the client // generating of dynamic web pages;

**Max 5**

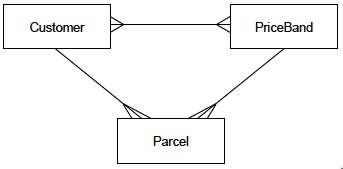
**[10]**

**Q7.**

(a)     **1 mark** for any one correct relationship drawn

**2 marks** for all three correct relationships drawn

**I.** Any additional writing on diagram



**2**

(b)     UPDATE PriceBand

SET Price = 5.99

WHERE ServiceSpeed = "Express"

    AND MinWeight = 1000

    AND MaxWeight = 4999

**1 mark** for UPDATE PriceBand

**1 mark** for SET Price = 5.99

**1 mark**\* for ServiceSpeed = "Express"

**1 mark**\* for either MinWeight = 1000 or MaxWeight = 4999 (or both joined by AND). **A.** use of >= and <= instead of = if conditions given for both MinWeight and MaxWeight.

To award both marks indicated by \* symbol, the conditions must be joined by ANDs.

**A.** Double or single quotes around Express

**A.** Express written in any case

**A.** £ symbol before 5.99

**A.** Table names before fieldnames

**DPT** for fieldname before table name.

**DPT** for unnecessary punctuation e.g. quotes where they should not appear. Allow one semicolon at the very end of the statement, but not at the end of each clause.

**DPT** use of incorrect equality operator e.g. ==

**4**

(c)     **Alternative 1**

SELECT DateSent, Postcode, ServiceSpeed, Price

FROM Parcel, PriceBand

WHERE CustomerID = 109

   AND Parcel.ServiceSpeed = PriceBand.ServiceSpeed

   AND Parcel.Weight >= PriceBand.MinWeight

   AND Parcel.Weight <= PriceBand.MaxWeight

ORDER BY DateSent

**Alternative 2**

SELECT DateSent, Postcode, ServiceSpeed, Price

FROM Parcel INNER JOIN PriceBand ON

Parcel.ServiceSpeed = PriceBand.ServiceSpeed

   AND Parcel.Weight >= PriceBand.MinWeight

   AND Parcel.Weight <= PriceBand.MaxWeight

WHERE CustomerID = 109

ORDER BY DateSent

**1 mark** for SELECT clause with correct four fields

**1 mark** for FROM clause with correct two tables

**1 mark** for CustomerID = 109

**1 mark** for Parcel.ServiceSpeed=PriceBand.ServiceSpeed

**1 mark** for Parcel.Weight >= PriceBand.MinWeight AND Parcel.Weight <= PriceBand.MaxWeight

**1 mark** for ORDER BY DateSent

**MAX 2** of the 3 marks for conditions if not joined by ANDs

Conditions linking the two tables can be present in either the FROM or WHERE clause or a mixture of both, as long as they are syntactically and logically correct.

Marks for correct files/tables in SELECT and FROM statements should not be awarded if additional fields/tables included, except allow the inclusion of the CUSTOMER table in the FROM statement so long as it has been correctly linked to the PARCEL table.

Marks can be awarded for the conditions in the WHERE statement even if the required tables are not present in the FROM.

**A.** Table names before fieldnames.

**A.** Use of Alias/AS command e.g. FROM Parcel AS P then use of P as table name (note some dialects of SQL do not require AS e.g. FROM Parcel P)

**A.** Insertion of spaces into fieldnames.

**A.** 109 with no delimiters or delimited using " or '.

**A.** Use of BETWEEN command for weight range e.g. Parcel.Weight BETWEEN PriceBand.MinWeight AND PriceBand.MaxWeight

**A.** ORDER BY written as one word ORDERBY.

**A.** ASC at the end of ORDER BY.

**I.** Unnecessary brackets.

**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.

**DPT** for fieldname before table name.

**DPT** use of incorrect equality operator e.g. ==

**Refer responses using nested SQL queries to team leaders.**

**6**

(d)     Parcel(ParcelID, ServiceSpeed, Weight, DateSent, CustomerID, RecipientName, HouseNumber, Postcode)

PostcodeLookup(Postcode, Street, Town, County)

**1 mark** for identifying that a new PostcodeLookup relation is required. Purpose must be clear; it is not sufficient to just make a new relation. Purpose could be made clear by any one of: appropriate name of relation (**A.** Address), approximately the correct attributes (allow, for example, incorrect inclusion of house number or CustomerID) in relation or having Postcode as the primary key.

**1 mark** for correct attributes in PostcodeLookup relation and identifying the Postcode as the primary key

**1 mark** for correct attributes left in Parcel relation and correct primary key

**A.** Answers given as SQL commands. As the question did not ask for this, perfect syntax is not required.

**A.** Alternative names for entities, so long as meaning is clear.

**A.** Spaces in entity and attribute names.

**A.** PostcodeLookup relation called Postcode even through this is the same as an attribute name.

**A.** Addition of unnecessary new relation for recipients which is not required for this question.

**R.** Do not award marks for correct attributes in a relation if additional attributes included.

**3**

**[15]**

**Q8.**

(a)     **Mark is for AO2 (apply)**

RaceEntryAndResult(RaceNumber, AthleteNumber, TimeSet)

**1 mark** for underlining both RaceNumber and AthleteNumber.

**1**

(b)     **All marks AO1 (knowledge)**

**Any 2 from:** Data is atomic / / no repeating groups (of attributes);  
**R** No repeated columns / attributes / data / values

No partial (key) dependencies / / No (non-key) attribute depends on part of the primary key but not the whole of it / / all non-prime attributes are (functionally) dependent on the whole of every candidate key / / (non-key) attributes depend on the whole key;

No non-key dependencies / / No transitive dependencies / / (non-key) attributes depend on nothing but the key;

Every (non-key) attribute is dependent upon the key;

Every determinant is a candidate key;

**A** ‘field’ for ‘attribute’

**A** ‘part’ for ‘partial’

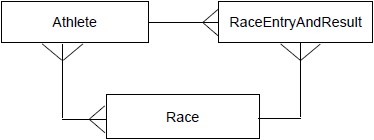
**MAX 2**

**2**

(c)     **All marks AO2 (analyse)**

**1 mark** for any one correctly drawn relationship **OR**

**2 marks** for all three relationships drawn correctly



**2**

(d)     **All marks AO3 (programming)**

**Method 1:**

INSERT INTO RaceEntryAndResult  
VALUES (6,27,"00:00.00")

**Method 2:**

INSERT INTO RaceEntryAndResult (RaceNumber, AthleteNumber, TimeSet)  
VALUES (6,27,"00:00.00")

**Method 3 (Default Time Assumed):**

INSERT INTO RaceEntryAndResult(RaceNumber, AthleteNumber)  
VALUES (6, 27)

**1 mark** for correct INSERT INTO clause

**1 mark** for correct VALUES clause

**A** default time delimited by any type of quotation mark or hashes or no delimiter

**A** any sensible variation on the default time eg "0:00", "00:00:00", or just 0

**A** the values 6 and 27 if they are delimited by any type of quotation mark

**A** list of fields in any order for method 2, but to get the VALUES mark in method 2, order of fields list in INSERT INTO must match order of values in VALUES

**2**

(e)     **1 mark for AO2 (analyse) and 2 marks for AO3 (programming)**

**AO2 (analyse) – 1 mark:**

**1 mark** for correctly identifying the table in the data model that needs to be updated (RaceEntryAndResult) and the conditions that should be used to identify the correct record to in the table to update – with both conditions linked by the correct logical operator

**Note:** The AO2 mark for understanding the data model should be awarded regardless of whether correct SQL syntax is used or not as they are for data modelling, not syntactically correct SQL programming

**AO3 (programming) – 2 marks:**

1 mark for correct SQL syntax in two of the three clauses (UPDATE, SET, WHERE)

**OR**

**2 marks** for correct SQL syntax in all three clauses – to get two marks, there must be fully correct SQL syntax and all three clauses must be present, but it might be possible that the AO2 mark was not awarded eg if OR was used instead of AND

Example Solution

UPDATE RaceEntryAndResult  
SET TimeSet = "00:18.76"  
WHERE AthleteNumber = 27 AND RaceNumber = 6

Additional Guidance

**AO3 marks:**

**A** any type of quotation marks or hashes for delimiters for TimeSet or no delimiters  
**A** the values 27 and 6 if they are delimited by any type of quotation mark  
**A** any sensible format for the time data eg "18.76", "18:76", "0:18:76" etc.

(f)      **3 marks for AO2 (analyse) and 2 marks for AO3 (programming)**

**AO2 (analyse) – 3 marks:**

**1 mark** for correctly understanding the data model and identifying the tables that data needs to be extracted from and the fields that need to be extracted, and including these and no other tables or fields in the query

**1 mark** for correctly identifying how the data in the required tables should be combined to produce the desired result (the linking condition)

**1 mark** for identifying the correct conditions to use within the model for the RaceNumber and TimeSet fields to retrieve the required data and for using the correct logical operators between all of the conditions

**Note:** The AO2 marks for understanding the data model should be awarded regardless of whether correct SQL syntax is used or not as they are for data modelling, not syntactically correct SQL programming

**AO3 (programming) – 2 marks:**

**1 mark** for correct SQL syntax in two of the four clauses  
(SELECT, FROM, WHERE, ORDER BY)

**OR**

**2 marks** for correct SQL syntax in all four clauses – to get two marks, there must be fully correct SQL syntax and all four clauses must be present, but there could be mistakes in the marks awarded for AO2 e.g. an incorrect or missing condition

Example Solutions

**Example 1**

SELECT AthleteNumber, Forename, Surname, TimeSet  
FROM Athlete, RaceEntryAndResult  
WHERE RaceNumber = 6  
AND TimeSet "00:00.00"  
AND Athlete.AthleteNumber = RaceEntryAndResult.AthleteNumber  
ORDER BY TimeSet

**Example 2**

SELECT AthleteNumber, Forename, Surname, TimeSet  
FROM Athlete INNER JOIN RaceEntryAndResult  
    ON Athlete.AthleteNumber = RaceEntryAndResult.AthleteNumber  
WHERE RaceNumber = 6  
AND TimeSet "00:00.00"  
ORDER BY TimeSet

Additional Guidance

**AO2 marks:**

Mark(s) can be awarded for the correct logical conditions even if the required tables are not identified as being used by the query  
Accept alternatives for not equal to that are correct in the context of the data model eg > or !=  
Accept any sensible variation on the default time eg "0:00", "00:00:00", or just 0  
Ignore unnecessary clause Race.RaceNumber = RaceEntryAndResult.RaceNumber

**AO3 marks:** Accept table names before fieldnames.  
Accept use of Alias/AS command eg FROM Athlete AS A or FROM Athlete A then use of A as table name.  
Accept INNER JOIN written as one word i.e. INNERJOIN.  
Accept ORDER BY written as one word i.e. ORDERBY.  
Accept ASC at end of ORDER BY clause.  
Accept insertion of spaces into fieldnames.  
Accept use of ", ' or # as delimiters for times.  
Accept use of " or ' as delimiters for around number 6.  
Ignore unnecessary brackets.  
**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.  
**DPT** for fieldname before table name.

**Refer responses using nested SQL queries to team leaders.**

**5**

(g)     **All marks AO1 (understanding)**

**Problem Conditions (1 mark):**

When two users try to update the same record simultaneously;

**How dealt with (2 marks):**

Alternative 1 - Record Locks

Maintain information about which records are currently being accessed;

When a user tries to access a record, consult this information and only permit access if record is not currently being used / / only permit read access to a record that is already open;

Alternative 2 - Transaction Queuing

Updates / database changes are (grouped as transactions and) queued;

Database software processes transactions in FIFO order from queue;

**Award 1 mark for ‘use of record locks’ if no other marks awarded for how dealt with.**

**3**

**[18]**

**Q9.**

(a)     Qualification(SubjectName, Level)  
Module(ModuleCode, ModuleName, UMSPoints, SubjectName, Level)  
Entry(CentreNumber, CandidateNumber, ModuleCode, ExamSession)

**1 mark** for naming all three relations correctly (Qualification, Module, Entry) − if alternative names are used, the purpose of the tables must be clear to award this mark.  
**1 mark** for one relation containing the correct attributes OR **2 marks** for all three relations containing the correct attributes.  
**1 mark** for correct primary key identified in one relation OR **2 marks** for correct primary key identified in all three relations.

**A** Alternative names for relations and attributes, so long as meaning is clear.  
**A** Spaces in relation and attribute names.  
**A** Introduction of a QualificationID field into the Qualification relation with it being used as the primary key (and therefore link to the Module table as well)  
**A** Introduction of an EntryID field into the Entry relation with it being used as the primary key.  
**A** ExamSession attribute split into parts e.g. Season, Year or just as a Date.  
**A** Creation of a fourth relation for sessions, containing a SessionID as the primary key and the ExamSession as the other attribute then the use of the SessionID in the Entry relation instead of ExamSession.  
**A** Accept the addition of other attributes to relations that are not asked for, as long as they are relevant e.g. ‘Extra Time’ in the Entry relation or ‘Entry Restrictions’ to the Qualification relation **but** a relation should not be considered to have the correct attributes if additional attributes are added that are irrelevant or would mean that the design is not normalised.  
**I** If any unnecessary relations are created for the first mark on relation names.  
**I** Any representation for foreign keys.  
**R** Attribute names that are the same as relation names.

**5**

(b)     Composite (key);  
**A** Compound (key)

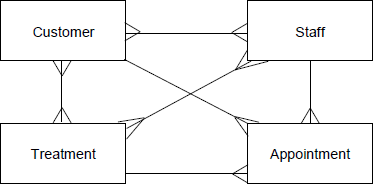
**1**

**[6]**

**Q10.**

(a)     **1 mark** for any one correct relationship drawn

**2 marks** for three correct relationships drawn



**MAX 1 if any incorrect relationships drawn**

**2**

(b)     TreatmentName VARCHAR(20) PRIMARY KEY  
//  
TreatmentName VARCHAR(20)  
PRIMARY KEY(TreatmentName)

Price SMALLMONEY  
TimeTaken INT  
NeedsQualification BOOLEAN

**1 mark** for TreatmentName, with sensible data type and identified as primary key

**1 mark** for two other fields with sensible data types and lengths (if given) *OR* **2 marks** for all three other fields with sensible data types and lengths

**A** Any sensible types. Lengths do not need to be specified.

**Valid alternative SQL types are:**

•        Alternative types for TreatmentName: char, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2, text, tinytext, mediumtext, longtext

•        Alternative types for Price: money, float, real, decimal, double, numeric, currency  
**R** integer only types for Price

•        Alternative types for TimeTaken: tinyint, smallint, mediumint, integer, number, byte, time, date / time

•        Alternative types for NeedsQualification: yes / no, bit, byte, bool, tinyint, enum(“yes”,“no”) − allow sensible alternative values in enum.

There should be a comma between the creation of each field, but ignore if these are missing, and accept a semi-colon at the end of the whole query.

**Answers using a syntax that is clearly not SQL should be awarded zero marks. But:**

•        **ignore one punctuation error e.g. unnecessary colons between fieldname and type**

•        **answers in SQL style syntax but using non-SQL data types can be credited but MAX 1 of 2 for other fields if any non-SQL types used.**

**3**

(c)     **Alternative 1**

SELECT EmailAddress, Forename, Surname  
FROM Customer, Appointment  
WHERE TreatmentName = "Luxury Manicure"

AND ApDate >= "01/01/2014"

AND ApDate <= "31/12/2014"

AND Customer.CustomerID =

Appointment.CustomerID

**1 mark** for SELECT clause with correct three fields  
**1 mark** for FROM clause with correct two tables  
**1 mark** for TreatmentName = "Luxury Manicure"  
**1 mark** for ApDate >= "01/01/2014"  
**1 mark** for ApDate = "01/01/2014"  
AND ApDate = "01/01/2014"  
**1 mark** for ApDate "31/12/2013", ApDate   
"01/01/2015" as alternative date criteria.  
Accept dates written in any format eg "12-31-2013"  
Valid delimiters for dates are ", ' or #  
Valid delimiters for strings are " or '  
Valid symbols between date parts are /, - or no symbol  
Ignore unnecessary clause

Appointment.TreatmentName =  
Treatment.TreatmentName

Ignore unnecessary brackets.  
Accept the following alternative methods for specifying the year, each of which are worth **2 marks**:

YEAR(ApDate) = 2014,

DATEPART("yyyy",ApDate) = 2014 or no quotation marks  
DATEPART("yy",ApDate) = 2014 or no quotation marks  
DATEPART("Year",ApDate) = 2014 or no quotation marks  
LIKE(ApDate, "\*2014\*") or "\*2014"  
LIKE(ApDate, "%2014%") or "%2014"  
LIKE "\*/\*/2014"  
LIKE "\*2014"  
BETWEEN "01/01/2014" AND "31/12/2014" or allow

"01/01/2015" as upper limit

**DPT** for unnecessary punctuation − allow one semicolon at the very end of the statement, but not at the end of each clause.  
**DPT** missing delimiters around data values, eg no quotation marks around dates.  
**DPT** for fieldname before table name.

**Refer responses using nested SQL queries to team leaders.**

**6**

**[11]**

**Q11.**

(a)     *Declaring PolicyNumber as primary key:*

PolicyNumber INT PRIMARY KEY(NOT NULL)

|  |  |
| --- | --- |
| / / PolicyNumber INT PRIMARY KEY(PolicyNumber) |  |

*Declaring RegistrationNumber as foreign key:*

RegistrationNumber CHAR(7) FOREIGN KEY REFERENCES Vehicle(RegistrationNumber)  
/ /  
RegistrationNumber CHAR(7)  
FOREIGN KEY (RegistrationNumber) REFERENCES Vehicle(RegistrationNumber)

*Declaring three other fields:*

DateStarted DATE  
PolicyType VARCHAR(13)  
ExcessAmount SMALLMONEY

**1 mark** for PolicyNumber with sensible type and length (if required), and identified as primary key. Type can be either numeric or text.

**1 mark** for two other fields from RegistrationNumber, DateStarted, PolicyType, ExcessAmount with sensible data types and lengths (if required by the type) *OR* **2 marks** for all four other fields with sensible data types and lengths (if required by the type)

•        Length of RegistrationNumber, if specified, must be 7.

•        Length of PolicyType, if specified, must be at least 13.

**1 mark** for identifying RegistrationNumber as a foreign key.

**MAX 3**

**Valid alternative SQL types are:**

•        Alternative types For *PolicyNumber*: smallint, mediumint, integer, any text field type (see below)

•        Alternative types For *DateStarted*: smalldatetime, datetime, datetime2, datetimeoffset

•        Alternative types For *PolicyType*: ENUM('Comprehensive', 'Third Party') - accept any type of quotation marks around values - accept data values in any order - accept if ENUM defined as a type separately first

•        Alternative types for *ExcessAmount*: money, currency, float, real, decimal, double, numeric, int, smallint, mediumint, integer

•        Alternative types for *text fields*: char, varchar, nchar, nvarchar, text, ntext, longvarchar, varchar2, nvarchar2, text, tinytext, mediumtext, longtext

**Sensible non-SQL data types can also be credited but MAX 2 if any non-SQL types used.**

**3**

(b)     UPDATE Vehicle  
SET Colour = "pink"  
WHERE RegistrationNumber = "DF24JUT"

**1 mark** per correct line

**A** double or single quotes around pink and DF24JUT

**A** table names before fieldnames

**A** pink written in any case

**DPT** no quotes

**DPT** for fieldname before table name

**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause

**MAX 2**

**2**

(c)     SELECT Model, Colour, Forename, Surname   
FROM Owner, Vehicle  
WHERE RegistrationNumber = "AB72XHC"  
AND Owner.OwnerID = Vehicle.OwnerID

**1 mark** for correct four fields in SELECT clause

**1 mark** for correct two tables in FROM clause

**1 mark** for WHERE RegistrationNumber = "AB72XHC"

**1 mark** for Owner.OwnerID = Vehicle.OwnerID, joined to other condition with AND

--- OR ---

SELECT Model, Colour, Forename, Surname  
FROM Owner INNER JOIN Vehicle ON Owner.OwnerID = Vehicle.OwnerID  
WHERE RegistrationNumber = "AB72XHC"

**1 mark** for correct four fields in SELECT clause

**1 mark** for correct two tables in FROM clause

**1 mark** for INNER JOIN using Owner.OwnerID = Vehicle.OwnerID

**1 mark** for WHERE RegistrationNumber = "AB72XHC"

Marks for SELECT and FROM statements should not be awarded if additional fields / tables included.

Accept table names before fieldnames.

Accept use of Alias / AS command eg FROM Vehicle AS V then use of V as table name.

Accept insertion of spaces into fieldnames

**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.

**DPT** for fieldname before table name.

**Refer responses using nested SQL queries to team leaders.**

**4**

(d)     (i)      Sequence of instructions / program / code;

**NE** programming language

**Note**: Do not award mark for program if candidate clearly means HTML

which is executed / run / interpreted on the server (instead of the client);

executed / run / interpreted when a web page is requested;

to generate a web page (and its contents) / result which the server returns to

the client / / generating of dynamic web pages;

**MAX 2**

**2**

(ii)     **1 mark for this point:**

Retrieve RegistrationNumber / value input by user and store in variable;

**R** responses that suggest the command makes the user input the values at the point in time when the script is run

**MAX 1 point from this list:**

from the web page / web site / form / web server / browser / url / request;

using POST / GET methods;

**2**

(iii)    Output the forename and surname;

Back to the web server / web browser / client / terminal;

**A** display forename and surname on web page (or alternative) for both marks

**R** responses that imply output is made directly to screen

**2**

(e)     Create a new table / / suitable table name given eg

SafetyCertificates;

with CertificateNumber as the primary key ;

Include these fields in new table: CertificateNumber, DateIssued, GarageName;

Add RegistrationNumber into the new table as a foreign key / / as link to Vehicle table;

**A** relation for table

**A** different fieldnames for new fields if meaning the same

**A** adding the extra field ExpiryDate, but not as an alternative to DateIssued

**A** answers by example eg writing out the new table definition, SQL script to achieve changes

**R** a composite key in new table

**Do not award any marks unless it is clear that a new table has been created**

**3**

**[18]**

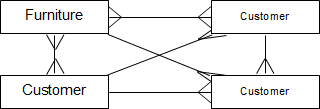
**Q12.**

(a)     **What means:**every attribute (in relation) is dependent on the key;  
the whole key;  
and nothing but the key;  
**R** Everything  
OR  
(relations) contain no repeating groups (of attributes) // data is atomic;  
no partial dependencies;  
no non-key dependencies;  
**R** No repeated columns / attributes / data  
OR  
every determinant (in the relation) is a candidate key;;  
**Max 2**

**Why important:**Eliminate update anomalies; **A** Example  
Eliminate insertion anomalies; **A** Example  
Eliminate deletion anomalies; **A** Example  
Eliminate data inconsistency // improve consistency // avoid inconsistency problems;  
\*Minimise data duplication // no unnecessaryrepeated data; **A** Reduce for minimise **R** eliminate  
\*Eliminate data redundancy; **A** Reduce / minimise for eliminate  
**NE** Easier to update / insert / delete without concrete example or good explanation  
**NE** Less errors whenupdating / inserting / deleting without concrete example or good explanation  
**NE** Saving space / memory  
**NE** Easier to query   
**Award marks to points made anywhere across (a)  
Can only award one of the two marks indicates by asterisks (\*)  
Max 2**

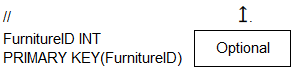
**4**

(b)     One mark per correct relationship.



**Max 2 if any incorrect relationships drawn  
Max 3**

**3**

(c)     FurnitureID INT PRIMARY KEY NOT NULL  


*Note that currency is not a valid SQL type*

FurnitureName VARCHAR(30)  
Category VARCHAR(10)  
Price SMALLMONEY  
SupplierName VARCHAR(20)

*Allow lengths after numeric types e.g. INT(11) as these are allowed in MySQL.*

**1 mark** for FurnitureID, with sensible data type and identified as primary key

**1 mark** for two other fields with sensible data types and lengths *OR* **2 marks** for all four other fields with sensible data types and lengths

**A** any sensible types. Lengths do not need to be specified.

**Valid alternative SQL types are:**

•        Alternative types For *FurnitureID:* smallint, mediumint, integer, any text type (see below)

•        Alternative types for *Price:* money, float, real, decimal, double, numeric, int, smallint, mediumint, integer

•        Alternative types for *text fields:* char, varchar, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2, text, tinytext, mediumtext, longtext

**Answers using a syntax that is clearly not SQL should be awarded zero marks. But:**

•        **ignore punctuation errors e.g. unnecessary colons or commas.**

•        **answers in SQL style syntax but using non-SQL data types can be credited but Max 1 of 2 for data types if any non-SQL types used.**

**3**

(d)     SELECT CustomerName, TelephoneNumber  
FROM Customer, CustomerOrder,   
CustomerOrderline  
WHERE FurnitureID=10765  
AND   
Customer.CustomerID= CustomerOrder.CustomerID  
AND CustomerOrder.OrderID= CustomerOrderLine.OrderID  
ORDER BY CustomerName (ASC)

**1 mark** for correct two fields in SELECT clause  
**1 mark** for correct three tables in FROM clause  
**1 mark** for FurnitureID = 10765  
**1 mark** for Customer.CustomerID = CustomerOrder.CustomerID,   
joined to other conditions with AND  
**1 mark** for CustomerOrder.OrderID = CustomerOrderLine.OrderID,  
joined to other conditions with AND  
**1 mark** for ORDER BY CustomerName, ASC is optional

                    --- OR ---

SELECT CustomerName, TelephoneNumber  
FROM Customer INNER JOIN CustomerOrder  
ON   
Customer.CustomerID=CustomerOrder.CustomerID INNER JOIN CustomerOrderLine ON  
CustomerOrder.OrderID=CustomerOrderLine.OrderID  
WHERE FurnitureID = 10765  
ORDER BY CustomerName (ASC)

**1 mark** for correct two fields in SELECT clause  
**1 mark** for correct three tables in FROM clause  
**1 mark** for INNER JOIN using Customer.CustomerID=CustomerOrder.CustomerID  
**1 mark** for INNER JOIN using CustomerOrder.OrderID=CustomerOrderLine.OrderID  
**1 mark** for FurnitureID = 10765  
**1 mark** for ORDER BY CustomerName, ASC is optional

Marks for SELECT and FROM statements should not be awarded if additional fields / tables included.  
Marks can be awarded for the conditions in the WHERE statement even if the required tables are not present in the FROM.  
Accept FurnitureID with no quotation marks, single quotation marks or double quotation marks.  
Accept table names before fieldnames.  
Accept use of Alias / AS command e.g. FROM Customer AS C then use of C as table name.  
Accept insertion of spaces into fieldnames  
Ignore unnecessary clause CustomerOrderLine.FurnitureID=Furniture.FurntiureID  
**I** unnecessary brackets  
**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.  
**DPT** for fieldname before table name.

**Refer responses using nested SQL queries to team leaders**

**6**

(e)     One mark for tick in correct row. Do not award mark if more than one row is ticked.

|  |  |
| --- | --- |
| **Command** | **Correct? (Tick One)** |
| ALTER TABLE | ✔ |
| CREATE FIELD |  |
| INSERT COLUMN |  |

**1**

**[17]**

**Q13.**

(a)     Composite (key);   
**A** Compound (key)

*Note: The word key is not required*

**1**

(b)     Data is atomic // no repeating groups (of attributes);   
**R** No repeated columns / attributes / data / values

No partial (key) dependencies // No (non-key) attribute depends on part of the primary key but not the whole of it // all non–prime attributes are (functionally) dependent on the whole of every candidate key // (non-key) attributes depend on the whole key;

No non-key dependencies // No transitive dependencies // (non-key) attributes depend on nothing but the key;

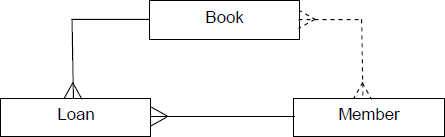
Every (non-key) attribute is dependent upon the key;

Every determinant is a candidate key;

**A** “field” for “attribute”  
**A** “part” for “partial”

**Max 2**

(c)



*1 mark per correct relationship (the dashed one is given)*

*Max 1 if more than two relationships drawn*

***2***

(d)     **Solution 1:**

SELECT EmailAddress, Forename, Surname   
FROM Book, Member, Loan  
WHERE Author = ‘Lucas Bailey’ AND  
               Book.BookID=Loan.BookID AND  
               Member.MemberID=Loan.MemberID

*1 mark for correct three fields in SELECT clause  
1 mark for correct three tables in FROM clause  
1 mark for Author = ‘Lucas Bailey’.  
1 mark for Book.BookID=Loan.BookID linked by AND  
1 mark for Member.MemberID=Loan.MemberID linked by AND*

**Solution 2:**

SELECT EmailAddress, Forename, Surname  
FROM Book INNER JOIN Loan ON Book.BookID=Loan.BookID  
          INNER JOIN Member on  
          Member.MemberID=Loan.MemberID  
WHERE Author = ‘Lucas Bailey’

*1 mark for correct three fields in SELECT clause  
1 mark for correct three tables in FROM clause  
1 mark for join from Member to Loan  
1 mark for join from Loan to Book  
1 mark for Author = ‘Lucas Bailey’  
Note: Joins do not need to be done in same order as example*

*Do not award mark for SELECT clause if extra attributes listed.  
Do not award mark for ‘Lucas Bailey’ unless it is enclosed in single or double quotation marks.****A*** *table names before fieldnames.****A*** *use of Alias / AS command e.g. FROM Member as M then use of M as table name.****A*** *insertion of spaces into fieldnames****DPT*** *for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause. Also, allow insertion of brackets at logically allowable places in the WHERE/FROM clauses.****DPT*** *for fieldname before table name.*

***Refer responses using nested SQL queries to team leaders.***

***5***

(e)     **Alternative 1:**INSERT INTO Book  
VALUES ( 837023, “Kenyan Safari”, “Karen Matu”, “African Travel Guides” )

**Alternative 2:**INSERT INTO Book (BookID, Title, Author, Publisher)  
VALUES (837023, “Kenyan Safari”, “Karen Matu”, “African Travel Guides” )

*1 mark for INSERT INTO Book;  
1 mark for correct field values. If alternative 2 is used, the order of the values and fieldnames must correspond to each other;*

*The values Kenyan Safari, Karen Matu and African Travel Guides must be in single or double quotation marks for the mark to be awarded.****A*** *the value 837023 with or without quotation marks.*

***A*** *Minor errors in transcribing the data from the question into the answer.****A*** *omission of brackets*

***2***

(f)     *One mark for principle and max two marks for implementation.*

**Principle:**

Create a new table (**A** link table) (BookCopy); through which Book and Loan tables will be (indirectly) linked;

**Implementation details using a new primary key:**

Create a new unique ID/key field (e.g. CopyID) (for each copy);  
Store the BookID and the CopyID in the new table;   
Replace the BookID in the Loans table with this CopyID;

*Note: In this implementation, CopyID is unique, i.e. BookID 1 and 2 cannot both have CopyID 1.*

**Implementation details using a composite key:**

Create a new field CopyID;  
Composite key formed by BookID and CopyID; **TO** if composite key is clearly in book table or loan table  
Store the BookID and the CopyID in the new table;   
**R** adding CopyID to Book table as this would created data redundancy but this does not talk out other mark scheme points  
Add the CopyID field to Loans table;   
**R** replace BookID with CopyID

*Note: In this implementation, CopyID is not unique, e.g. BookID 1 and 2 can both have CopyID 1.*

*Marks can be awarded for principle and/or implementation details.*

**A** Relation for Table  
**A** Answers if candidates have rewritten new relations, awarding marks where the points above can be observed in the redrawn relations;  
**A** alternative name for CopyID

**Max 3**

(g)     (i)      So that searching, adding and deleting can be done efficiently // To speed up searching, adding and deleting;  
**A** just one of searching, adding, deleting  
**NE** organise efficiently  
**NE** easily for efficiently

**1**

(ii)     **Alternative 1 (context-specific):**

A function/calculation that computes a record position/address; within a specified range; from a key field value;  
**A** an example of a hashing function e.g. calculate an integer from certain letters in a field for one mark

**Alternative 2 (generic):**A function (**A** algorithm) H, applied to a key k; which generates a hash value (H(k)) (of range smaller than the domain of values of k);

**Max 2**

(iii)     **What is** *(1 mark)*:  
When more than one key value maps to the same record position/address // when two keys compute the same hash value;  
**A** “two records”, “two items” or “two pieces of data“ for “two keys” but **R** “two files” – both in this question part only

**How dealt with** *(1 mark):*Store the record in the next available location in the file // store a pointer (in each file location) that points to a list of records that have all collided at the file location;  
**A** idea that each storage location could store more than one record e.g. five records per location, if explained.  
**A** example of what “next available” might be   
**A** key is rehashed

**A** table for file

**2**

**[20]**

**Q14.**

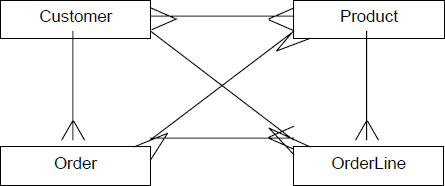
(a)     Only one (type of) product per order // Must make new order for each (type of)   
product; as ProductNumber / product details stored in relation that has   
OrderNumber as primary key / product relation directly related to order relation // as relations not (fully) normalised;

Difficult to query // requires (unnecessarily) complex queries; as contains repeating groups (of attributes);   
**A** either way round

**A** table for relation

**Max 2**

(b)



*1 mark for each correct relationship, up to Max 3  
Max 2 if more than three relationships drawn*.

**Max 3**

(c)     ProductNumber INTEGER PRIMARY KEY//

|  |  |
| --- | --- |
| ProductNumber INTEGER PRIMARY KEY(ProductNumber) |  |

ProductPrice SMALLMONEY  
ProductDescription VARCHAR(50)  
QuantityInStock INTEGER

*1 mark for ProductNumber correct with appropriate type and identified as primary key  
1 mark for two other fields correct with appropriate types OR 2 marks for all three other fields correct with appropriate types*

**A** any sensible types / field lengths. eg:  
For ProductNumber: integer, numeric, char, varchar, text, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2  
For ProductPrice: smallmoney, money, currency, float, real, decimal, dec, double, double precision, numeric  
For ProductDescription: varchar, char, varchar, text, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2  
For QuantityInStock: integer, numeric, float, real, decimal, dec, double, double precision, numeric

**A** insertion of other unnecessary but valid SQL commands e.g. AUTO INCREMENT, NOT NULL  
**I** Spaces inserted into fieldnames e.g. Product Number

*Max 2 if additional fields added*

**3**

(d)     Sequence of instructions / program / code;  
**NE** programming language  
  
*Note: Do not award mark for program if candidate clearly means HTML*which is executed/run/interpreted on the server (instead of the client); executed/run/interpreted when a web page is requested; [to generate a web page (and its contents) which the server returns to the client // generating of dynamic web pages;

**Max 2**

(e)     (i)      Max 1 point from this list:  
Retrieve ProductNumber and Quantity // retrieve values input by user;  
stores values in variables;  
**R** responses that suggest these two commands are making the user input the values

Max 1 point from this list:  
from the web page/web site/form/web serve/browser;  
using POST/GET methods;

**Max 2**

(ii)     Query/retrieve data from the products table;  
to retrieve the price of product being ordered/selected on form/product that has correct product number/product number in ProdNum;  
Store the set of records/data/price returned in ProdDetails;

**Max 2**

(iii)     To send/output the Total Price back to the web server / web browser / client;  
**A** display price on web page  
**R** sent to user / customer

**1**

(f)      **Either**

SELECT ProductNumber, ProductDescription, ProductPrice, Quantity  
FROM Product, OrderLine  
WHERE OrderNumber = 4013   
        AND Product.ProductNumber = OrderLine.ProductNumber  
ORDER BY ProductNumber ASC

*1 mark for SELECT clause with correct four fields  
1 mark for FROM clause with correct two tables  
1 mark for OrderNumber = 4013  
1 mark for clause linking tables on the common field with no additional unnecessary clauses added  
1 mark for ORDER BY ProductNumber, ASC is optional*

**Or**

SELECT ProductNumber, ProductDescription, ProductPrice, Quantity  
FROM Product INNERJOIN OrderLine ON  
  
Product.ProductNumber = OrderLine.ProductNumber  
WHERE OrderNumber = 4013  
ORDER BY ProductNumber ASC

*1 mark for SELECT clause with correct four fields  
1 mark for correct two tables in FROM clause  
1 mark for INNERJOIN together with ON   
Product.ProductNumber = OrderLine.ProductNumber and no other joins  
1 mark for OrderNumber = 4013  
1 mark for ORDER BY ProductNumber, ASC is optional*

***In both solutions:***

*Do not award mark for SELECT clause if extra attributes listed.  
Do not award mark for FROM clause if extra tables listed.  
Do not award mark for ORDER BY clause if order descending.  
Only award two marks for conditions if they are connected by AND.  
Otherwise just award one of the marks.  
If candidate appears to have written two queries e.g. there are two SELECT commands then mark the first query.***A** table names before fieldnames. i.e. TableName.FieldName  
**A** “ or ’ as delimiters for 4013  
**A** ascending, (ASC) for ASC  
**R** if ASC written before ProductNumber in ORDER BY  
**I** Spaces inserted into fieldnames e.g. Product Number  
**A** answers that candidates have surrounded by “ExecuteSQL()”.

If any of the errors listed below are made, they should result in at most one mark being lost. If the mistake is made more than once then on subsequent occasions, providing that the meaning is clear, the mistake should be ignored:

•        the addition of unnecessary punctuation such as semicolons

•        the fieldname being written before the tablename

**5**

**[20]**

**Q15.**

(a)     **What means:**   
every attribute (in relation) is dependent on the key; the whole key and nothing but the key;  
OR  
(relations) contain no repeating groups (of attributes) // data is atomic; no partial dependencies; no non-key dependencies;  
**R** No repeated columns/attributes/data  
OR  
every determinant (in the relation) is a candidate key;;

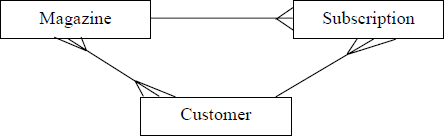
*Max 2*

**Why important:**Eliminate update anomalies; **A** Example **R** Easy to update **NE**Eliminate insertion anomalies; **A** Example  
Eliminate deletion anomalies; **A** Example  
Eliminate data inconsistency // improve consistency // avoid inconsistency problems;  
Minimise data duplication; **A** Reduce for minimise **R** eliminate  
Eliminate data redundancy; **A** Reduce/minimise for eliminate **A**No unnecessarily repeated data **R** No repeated data  
**R** Saving space/memory **NE**

*Max 2*

**4**

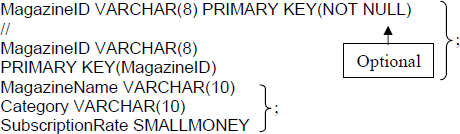
(b)



*1 mark for per correct relationship***I** incorrect relationships

**2**

(c)



*1 mark for MagazineID, and identified as primary key  
1 mark for other fields with appropriate data types***A** any sensible types / field lengths. Some examples are:  
For MagazineID: integer  
For SubscriptionRate: money, currency, float, real, decimal, dec, double, double precision, numeric  
As alternative to varchar: char, varchar, text, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2  
**R** answers clearly written in a different programming language

**2**

(d)     SELECT CustomerName, Address, Postcode  
FROM Magazine, Subscription, Customer  
WHERE MagazineName = ‘AQA Computing Now’  
          AND EndDate < ‘01/06/2010’AND  
          AND  
          Magazine.MagazineID = Subscription.MagazineID  
          AND  
          Subscription.CustomerID = Customer.CustomerID

*1 mark for SELECT clause with correct three fields (allow any additional fields from relations or \*)  
1 mark for FROM clause with correct three tables  
1 mark for MagazineName = ‘AQA ComputingNow’  
1 mark for EndDate   
1 mark for two clauses linking tables on the common field  
Max 1 of the 3 marks for conditions if not joined by ANDs*

OR

SELECT CustomerName, Address, Postcode  
FROM Magazine INNERJOIN Subscription ON  
  Magazine.MagazineID=Subscription.MagazineID INNERJOIN  
  Customer ON Subscription.CustomerID=Customer.CustomerID  
WHERE MagazineName = ‘AQA ComputingNow’  
          AND EndDate

*1 mark for SELECT clause including correct three fields (allow any additional fields from relations or \*)  
1 mark for correctly joining two tables in FROM clause  
1 mark for correctly joining the third table in FROM clause  
1 mark for MagazineName = ‘AQA ComputingNow’  
1 mark for EndDate*

***In both solutions:***Do not award mark for ‘AQA Computing Now’ unless it is enclosed in single or double quotation marks.  
For EndDate, accept # symbols or no delimiting symbols.

**A** EndDate day and month without preceding 0, i.e. 1 / 6.  
**A** <= ‘31/05/2010’ for EndDate.  
**A** table names before fieldnames.  
**A** use of Alias/AS command e.g. FROM Magazine as M then use of M as table name.  
**A** insertion of spaces into fieldnames  
**DPT** for unnecessary punctuation – allow one semicolon at the very end of the statement, but not at the end of each clause.  
**DPT** for fieldname before table name.

**5**

(e)     UPDATE Magazine  
SET MagazineName= ‘AQA Garden News’  
WHERE MagazineName= ‘AQA Gardening Monthly’  
  
*1 mark per correct line***A** double or single quotes around magazine names   
**R** no quotes  
**A** table names before fieldnames.  
**DPT** for fieldname before table name.

**Max 2**

**[15]**

**Q16.**

(a)     collection of **tables / relations;**

created through **common attributes / shared attributes;**

**A** common fields / shared fields

**A** primary keys and foreign keys

(must have both primary & foreign)

**2**

(b)     (i)      to uniquely identify each record  
/ to uniquely identify a particular instance of an entity;

**A** to uniquely identify a member

**1**

(ii)      because people can change their email addresses;

**1**

(iii)     to speed up searching for a particular item;

**1**

(iv)    MemberID;

**1**

(c)     (i)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field name | ItemOnOffer | Description | MemberID | Email address |
| Table | Item | Item | Item /Member | Member |
| Criterion | \* bed \* |  |  |  |
|  | **;;** | **;** | **;** | **;** |

 \* bed \* as criterion in ItemOnOffer – i.e. correct use of wild card

**2**

Or: bed/ childs bed/ bunk beds, as criterion for ItemOnOffer

**1**

Or no criterion in ItemOnOffer when \* bed \* in Description

**1**

 \* bed \* or \* bunk beds \* as criterion in Description

**2**

Or: no criterion in Description

**1**

MemberId, in table item / table member & no criterion

**1**

emailAddress in table Member & no criterion

**1**

**Max 3**

(c)     (ii)      3 *(if wild card used in either ItemOnOffer or Description)*;

2 *(If \* bunk bed \* in Description)*

*1 (If criterion for ItemOnOffer ‘bed’, ‘child’s bed’ or ‘bunk beds’*

**1**

**[10]**

**Q17.**

(a)     (i)      member ID / user name; password/PIN;

**A** account name instead of memberID;

**A** answers to security questions;

**2**

(ii)     Member (MemberID, CreditCardNo, Member(Full)Name, Address, DrivingLicenceNo, EmailAddress, Mobile(Tel)No/TelNo); + attributes from b(i)

**I** bars over attributes

**1**

(iii)     ParkingArea (LocationCode, ParkingAreaName, PostCode);

**A** ParkingAreaID instead of LocationCode

**R** ParkingArea   
**R** Name as attributes

**1**

(iv)    Car (CarRegNo, LocationCode);

**A** RegNo/CarReg instead of CarRegNo

Allow follow through on foreign key from (iii)

**1**

(v)     Booking (BookingRefCode, CarRegNo, MemberID, StartDateTime,  
EndDateTime, LocationCode);;;

*1 mark for CarRegNo and MemberID;*

*1 mark for StartDateTime and EndDateTime;*

*1 mark for LocationCode;*

*1 mark for BookingRefCode as primary key;*

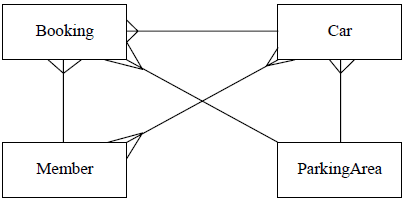
**A** 2 separate attributes for DateTime

**A** BookingRef/BookingID instead of BookingRefCode

Follow through on attribute names

**Max 3**

(b)



*1 mark for each correct relationship,*

*If 4 or 5 relationships given, mark as follows:*

*All 4/4 or 5/5 correct: 3 marks*

*3/4 or /54 correct: 2 marks*

*2/4 or3/5 correct: 1 mark*

*All other cases: 0 marks*

**I** *relationship between Member and Parking Area*

**Max 3**

(c)     SELECT MemberID, (MemberFullName,) CarRegNo,  
StartDateTime, (EndDateTime) FROM (Member,) Booking

**1**

WHERE Member.MemberID = Booking.MemberID

**1**

AND EndDateTime BETWEEN 1/12/07 AND 31/12/07

**1**

ORDER BY MemberID (ASC/DESC)

**1**

**A** other attributes if present in candidate’s booking table

**1**

**Alternative Answer:**

SELECT \*; FROM Booking;

WHERE EndDateTime LIKE “\*/12/07”

**A** StartDateTime instead of EndDateTime

**P1** if attribute.table notaion used

**P1** for extra punctuation or tbl in front of table name

**I** punctuation around dates/times

**I** case of keywords etc

**A** other wildcard characters

**Alternative Answer:**

SELECT MemberID, MemberFullName, CarRegNo,

StartDateTime, EndDateTime

FROM Member INNER JOIN Booking

ON Member.MemberID = Booking.MemberID

WHERE EndDateTime >= 1/12/07 AND  
 EndDateTime <= 31/12/07

ORDER BY MemberID

**Max 4**

**[15]**

**Q18.**

CREATE TABLE // CREATE INDEX // CREATE DOMAIN // CREATE TRIGGER // CREATE VIEW // GRANT …;  
**R** CREATE DATABASE

**1**

**[1]**

**Q20.**

(a)     (i)      Recipe table;

**A** Figure 2;

**1**

(ii)     **Why:** contains multiple values in Ingredients field/attribute/column

// data in Ingredients column not atomic // repeating groups;

**1**

(b)     (i)      **Fully normalised:**

Every attribute is dependent on the key, the whole key and nothing but the key;;

**OR** (tables contain no repeating groups of attributes,) no partial dependencies;

No non-key dependencies;

**A** rely on instead of depend on

**OR** if (and only if) every determinant in the relation is a candidate key;

**2**

(ii)     **Why:** to aid consistency of data // to avoid potential data inconsistency problems

// to eliminate data inconsistency // to minimise data duplication

// to eliminate data redundancy; A reduce instead of eliminate

**R** saving space

**1**

(c)     (i)      Recipe (RecipeID, Dish, PrepTime, CookTime, NoOfServings, CookInstructions);

**1**

(ii)     FoodItem (FoodItemID, FoodItemName, PackSize, Price);

**1**

(iii)     RecipeIngredient(FoodItemID, RecipeID, Quantity)

**4**

*1 mark for each correct field, 1 mark for correct primary key*

*(take off 1 mark for every extra field included)*

(d)     SELECT FoodItemName, Quantity, PackSize, Price (1)

FROM FoodItem, RecipeIngredient, Recipe (1)

WHERE (Recipe.RecipeId = RecipeIngredient.RecipeId) (1)

AND (RecipeIngredient.FoodItemId = FoodItem.FoodItemId) (1)

AND (Recipe.Dish = “Feta Salad”) (1)

ORDER BY FoodItemName ASC (1)

Field names **F/T**

**P1** for fieldname.tablename

**P1** tbl prefix

**A** ORDER BY FoodItemName

**A** Dish instead of Recipe.Dish

**A** ‘feta salad’ instead of ‘Feta Salad’ A #feta salad# instead of ‘Feta Salad’

**Max 5**

**[17]**

**Q21.**

(a)     By common / shared attributes; / by primary and foreign keys;

**A** actual example(s) from the tables

**A** fields instead of attributes

**1**

(b)     BookID, DateOut;;

**I** MemberID

*Other fields, penalise*

**2**

(c)     To speed up searching;

**1**

(d)     Check digit;

**1**

**[5]**

**Q22.**

(a)     Copyright, Designs and Patents Act (1998); *if other laws included* **T.O.**

**1**

(b)     Boxes for correct entities: SoftwareLicence SoftwareInstallation *one mark*

*Correct degree of relationship: 1 to many one mark*

*Suitable name for relationship: one mark*

**

**3**

(c)     Any sensible field length accepted except for SoftwareID, ComputerID, StaffID

(i)      SoftwareID VARCHAR(10) PRIMARY KEY (NOT NULL)

// SoftwareID VARCHAR(10)

PRIMARY KEY(SoftwareID);

*Could appear at end of list. It doesn.t have to be with softwareID VARCHAR(10) In fact, this would provide a syntax error*

SoftwareName VARCHAR(30)

Supplier VARCHAR(20)

DatePurchased DATE

Version VARCHAR(10)

ExpiryDate/DateValidTo DATE

NoOfLicences INT

*1 mark for any 3 attributes correct*

***P1*** *if extra symbols used*

*Ignore spaces and case in attribute*

**3**

(ii)     SoftwareID VARCHAR(10)

**A** char/string/text/alphanumeric instead of VARCHAR

**A** Date/Time instead of Date

**A** Integer instead of INT

*BOD any attributes which are clearly more than 1 word*

ComputerID VARCHAR(6)

DateInstalled DATE

StaffID VARCHAR(3)

PRIMARY KEY (SoftwareID, ComputerID);

FOREIGN KEY (SoftwareID)

REFERENCES Software Licence(SoftwareID);

*1 mark for any 2 attributes correct*

*If not DDL give 1 mark If composite key identified*

**I** NOT NULL

**4**

(d)     SELECT ComputerID, SoftwareName, Version ;

*Extra attributes: T.O.*

FROM SoftwareLicence, SoftwareInstallation ;

WHERE SoftwareLicence.SoftwareID=SoftwareInstallation.SoftwareID ;

ORDER BY ComputerID;

**A** ASC or DESC

*Accept (instead of FROM WHERE): FROM SoftwareLicence INNER JOIN SoftwareInstallation ON SoftwareLicence.SoftwareID = SoftwareInstallation.SoftwareID*

*P1 for other spurious punctuation inc semicolons*

**A** LEFT JOIN

*Table names prefixed with tbl,* ***P1***

*If table name and attribute transposed,* ***P1***

**4**

**[15]**

**Q23.**

(a)     **Disadvantages of system**

**Causes**

Repetitive data entry;

Data items stored on more than one file / duplicated;

Files cannot be shared;

No centralised, authoritative store of data;

**Effects:**

Wastes time;

Increases risk of errors;

Wastes storage space;

Can cause data inconsistency;

Won’t know which data is correct if two different versions;

*1 mark per cause and 1 per resultant effect to max*

*Can give more than 1 effect per cause*

**4**

(b)     **Sharing files**

Data files are structured differently / have different fields;

For different applications;

So may not be compatible;

//Different files may have e.g. names and addresses;

In different formats / field lengths;

**2**

(c)     **Definitions**

(i)      Attribute – a property or characteristic of an entity;

**1**

(ii)     Primary key – an attribute that will identify a particular instance of an entity  
**A**a field which identifies a record

**1**

(iii)     Foreign key – an attribute in one table that is (linked) to a primary key attribute in anther table;

**A** a field in one table which is a primary key field in a (linked) table

**1**

(d)     (i)      CustomerID;

**1**

(ii)     OrderNumber & OrderLineNumber

**A**OrderNumber & PartNumber

**1**

(iii)     PartNumber, OrderNumber

**1**

(iv)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Surname | FirstName | DateOfOrder | Description | Quantity |
| *Table* | Customer | Customer | Order | Stock | OrderLine |
| *Show* |  |  |  |  |  |
| *Criteria* | [(Enter) Surname] | [(Enter) First Name] | [(Enter) Date of Order] |  |  |
| *//Criteria* | Smith | Jeremiah | =23/04/06 |  |  |

**Mark as follows:** *nb Either Enter or [ ]*

**2nd column**Customer + [Enter Surname] / Smith

*1 mark*

***3rd Column****Customer + [Enter First Name] /Jeremiah*

*1 mark*

***4th Column****Order + [Enter Date of Order] or =23/04/06*

*//Order + = #23/04/06#*

***A*** *date in other similar formats, Order + 23/04/06*

*1 mark*

***5th Column****Description + Stock*

*// PartNumber + OrderLine*

*1 mark*

***6th Column****Quantity + OrderLine*

*1 mark*

*‘Show’ (tick, ‘Yes’, ‘true’) for Description (PartNumber) + Quantity*

*1 mark*

***A*** *Show in all columns*

***I*** *anything else in ‘show’ cells*

*(5th & 6th columns can be interchanged)*

***6***

***[18]***

**Q24.**

(a)     (i)      A HardwareItem

B EquipmentLoan

C ‘is out on’;

**A** any wording with similar meaning

**R** one to many relationship

*1 mark for A and B, 2 marks for C*

**2**

(ii)     Entity-Relationship Diagram;  
**A** E-R diagram

**A** E-R D         
**R** E-A-R diagram

**1**

(b)     CREATE TABLE Hardware Item

(Description VARCHAR (30)

Make VARCHAR(15)

Model VARCHAR(15)

**A** text/string instead of char/varchar

*1 mark*

(Inventory)RefNo CHAR(20) PRIMARY KEY,

**A** string/text/character/VARCHAR (20) instead of CHAR(20)

*1 mark*

PurchaseDate DATE,

**A** DateOfPurchase DATE  
**A**  Date/Time instead of Date

PurchasePrice CURRENCY,

Location VARCHAR(4))

**A** DECIMAL/MONEY/Number/Real/Float/Single instead of CURRENCY

**A** Room VARCHAR(4)

**A** INT/number instead of VARCHAR

*1 mark*

**Alternative for InventoryRefNo:**

(Inventory)RefNo CHAR(20), PRIMARY KEY(InventoryRefNo),

(Inventory)RefNo CHAR(20), NOT NULL,, PRIMARY KEY(InventoryRefNo),

**A** VARCHAR(20) instead of CHAR(20)

*Note: string lengths do not have to be exact/present except for InventoryRefNo*

CREATE TABLE EquipmentLoan

(Inventory)RefNo VARCHAR(20),

**A** NOT NULL

*If not DDL but composite key identified, give 1 mark*

Location VARCHAR(4),

(Staff)Initials VARCHAR(3),

DateRemoved DATE,

DateReturned DATE,

**A** NOT NULL

*1 mark*

PRIMARY KEY (InventoryRefNo, DateRemoved)

*1 mark*

FOREIGN KEY (InventoryRefNo) REFERENCES HardwareItem (InventoryRefNo))

*1 mark*

***P1*** *for extra attributes*

**6**

(c)     SELECT (HardwareItem.)Description, (EquipmentLoan.)DateRemoved,

*1 mark*

EquipmentLoan. (Inventory)RefNo,

**A** HardwareItem.InventroryRefNo

*1 mark*

FROM HardwareItem, EquipmentLoan

*1 mark*

WHERE HardwareItem. (Inventory)RefNo = EquipmentLoan.(Inventory)RefNo

*1 mark*

AND (EquipmentLoan.)DateRemoved > givenDate

**A** > =  
**A** = >

*1 mark*

ORDER BY (EquipmentLoan.) (Inventory)RefNo;

*1 mark*

**or**

SELECT (HardwareItem.)Description, (EquipmentLoan.)DateRemoved,

*1 mark*

EquipmentLoan. (Inventory)RefNo

**A** HardwareItem.InventroryRefNo

*1 mark*

FROM HardwareItem

INNER JOIN EquipmentLoan

*Note: can swap tables*

*1 mark*

ON HardwareItem. (Inventory)RefNo = EquipmentLoan. (Inventory)RefNo

*1 mark*

WHERE (EquipmentLoan.)DateRemoved > givenDate

**A** > =  
**A** = >

*1 mark*

ORDER BY (EquipmentLoan. )(Inventory)RefNo;

**A** HardwareItem.InventroryRefNo

*1 mark*

**F/T** with attribute names

**P1** for tbl prefix

**P1** if table name after attribute name

**I** extra punctuation

**6**

**[15]**

**Q25.**

(a)     CandidateNumber;

**1**

(b)     Table contains repeating groups;

**R** repeated data/fields/attributes

ModuleCode, ExamSession, ModuleMark, Level, TotalMark, Grade contain multiple values; *mention at least one attribute by name (forename/ surname T.O.)*

*There is redundant data* ***T.O.***

**Max 1**

(c)     1 mark for correct primary key, 1 mark for correct other attributes,

**I** spaces/underscores in attribute names

*Extra attributes =* ***T.O.***

(i)      Pupil (PupilForenames, Pupil Surname, CandidateNumber);

**A** (Forename,Surname,CandidateNo)

**2**

(ii)     ModuleResult (CandidateNumber, ModuleCode, ExamSession,

ModuleMark)

**2**

(iii)     PupilGrade (CandidateNumber, Level, TotalMark, Grade)

**2**

(d)



**2**

(e)     Must use same attributes as in (c) above (mark as F. T.)  **I** case

SELECT PupilForenames,Pupil Surname, Grade *I pupil*. / *pupilgrade*.(1)

FROM Pupil, PupilGrade(1)

WHERE Pupil.CandidateNumber = PupilGrade.CandidateNumber

AND Level=“ A”

*accept* Level=‘A’ *or* Level=A(1)

ORDER BY TotalMark DESC;      **A** Descending (1)

OR

SELECT PupilForenames,PupilSurname, Grade(1)

FROM Pupil INNER JOIN PupilGrade ON Pupil.CandidateNumber = PupilGrade. CandidateNumber(2)

WHERE Level = “A”

*accept* Level = ‘A’ *or* Level=A(1)

ORDER BY TotalMark DESC;       **R** = Desc(1)

*If pupilForename.pupil penalise once*

**5**

**[15]**

**Q26.**

(a)     Contains a repeating group;

**OR**

Cells for one or more of SubjectID, SubjectName,

ExamBoardSubjectOfficerName,

NumberOfCandidatesEntered contain multiple values;

**R** Repeating attributes, etc

**1**

(b)     Attribute names must not be redefined (exception: allow Center).

*1 mark for attributes(lose this mark if extra attributes), one mark for correct primary key*

*(i)     *

***2***

*(ii)    *

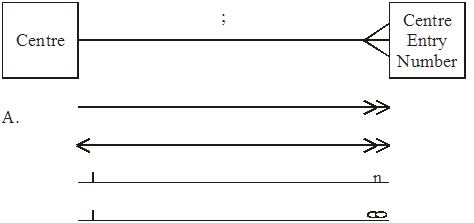
***2***

*(iii)   *

***2***

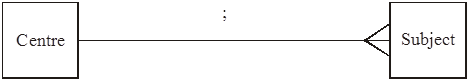
*Penalise misspelling once****I*** *spaces in attribute names****I*** *capitalisation*

(c)     (i)



**1**

(ii)



**1**

(d)     **I**. Inner join, Join

**A** Without commas

**R** Brackets in Select and anywhere else except:

**A** Brackets around (DateReported < 1/3/2005) and (ExaminationOfficer.CentreNo=Problem.CentreNo) as shown

Asc is optional but if present it must be at end of **Order By** line

(**A** Ascending)

*Penalise brackets once*

**

**Max 5**

(e)     **R** Brand names word processor//word processor with e-mail support;

**1**

**[15]**