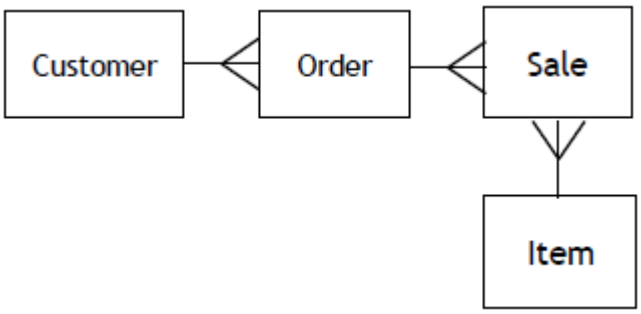
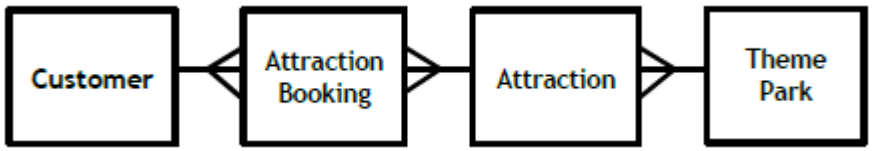
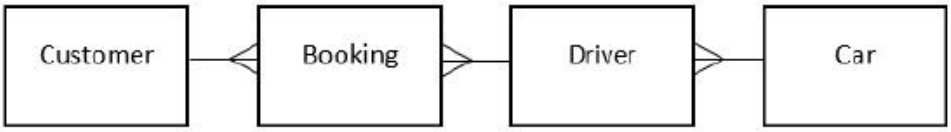
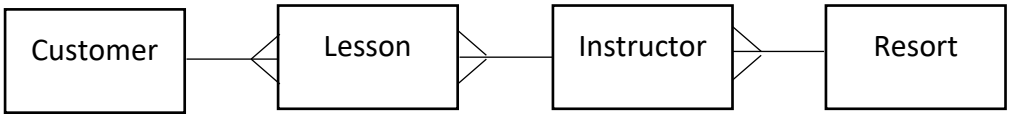
HIGHER COMPUTING SCIENCE **DATABASE DESIGN AND DEVELOPMENT**

PAST PAPER QUESTIONS

MARKING SCHEME

1a	Many-to-Many	**1**
1b	One-to-One	**1**
2a	A compound key is a primary key with more than one attribute to create a unique identifier.	**1**
2b	Order No and Item ID	**1**
2c		**3**
3	Boolean	**1**
4a	No single field provides a unique value, or All three fields are required to provide a unique value.	**1**
4b	Attribute name, Attribute type, Keys/PK/FK, Validation, Field length/size, Format, Required, Unique, Sample data, Entity name.	**2**
5		**3**
6a		**3**
6b	Customer.Known As, Booking.Booking ID, Booking.From, Booking.To, Booking.Cost Car.Registration OR Driver.Registration	**3**
6c	WHERE Booking.Booking ID=12345	**1**

6d SELECT Customer.[Known As], Booking.[Booking ID], Booking.[From], Booking.[To], Booking.[Cost], Driver.[Registration]
FROM Customer, Booking, Driver
WHERE Customer.[Customer ID] = Booking.[Customer ID] AND Driver.[Driver ID] = Booking.[Driver ID] AND Booking.[Booking ID] = 12345; **4**

7a 

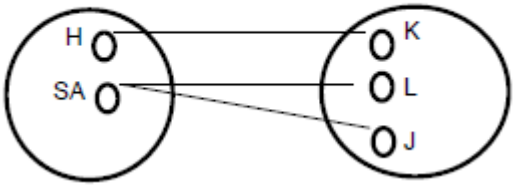
3

7b A compound key made up of InstructorID, StartTime, Date. **1**

7c SELECT Lesson.[Date], Instructor.{Instructor ID], Instructor.[FirstName], Customer.[FirstName], Customer.[Surname], Lesson.[StartTime], COUNT(Customer.[FirstName]) AS "Number of Lessons"
FROM Lesson, Instructor, Customer
WHERE Customer.[Customer ID] = Lesson.[Customer ID] AND Instructor.[Instructor ID] = Lesson.[Instructor ID] AND Instructor.[Instructor ID] = 14
GROUP BY Lesson.[Date], Instructor.{Instructor ID], Instructor.[FirstName], Customer.[FirstName], Customer.[Surname], Lesson.[StartTime]; **5**

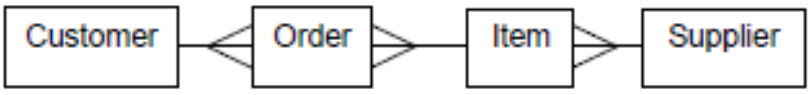
7d The COUNT function. **1**

8a School Pupil **2**



8b One-to-many **1**

9a **6**



(1 mark)for each correct join with no extra join and 1 more mark if the join has the correct cardinality

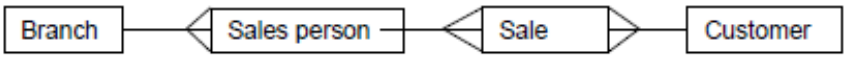
9b **5**

A – Integer	*(1 mark)*
B – Real	*(1 mark)*
C – Object/Link	*(1 mark)*
D – Lookup from Supplier	*(1 mark)*
E – N	*(1 mark)*

10a It can/may change (1 mark) not unique (1 mark) **2**

10b Difficult to search/sort (1 mark) for town/street/post code (1 mark) **2**

11a **6**



- 2 marks for each correct relationship without an extra relationship
- 1 mark for each relation with incorrect cardinality without an extra relationship
- 1 mark for each correct relationship with an extra relationship

11b Sale – Date (1 mark), amount (1 mark) **5**
Customer – Customer Name (1 mark)

11c SUM **1**

12a SELECT Customer.[Customer name], Customer.[Customer address], Order.[Order no], **3**
Order.[Date], Item.[Description], Item.[Cost], Sale.[Quantity]
FROM Customer, Item, Order, Sale
WHERE Customer.[Customer ID] = Order.[Customer ID] AND Item.[Item ID] =
Sale.[Item ID] AND Order.[Order no] = Sale.[Order No] AND Order.[Order no] = 10728;

12b SELECT SUM([Cost]*[quantity]) AS “Total” **2**

13a SELECT [Price (£)] * [Quantity] AS “Cost” **2**

13b SELECT SUM([Price (£)] * [Quantity]) AS “Total” **2**

14 SELECT username, COUNT(*) AS [Total Messages Made] **3**
FROM Message
GROUP BY username;

15 SELECT Title **2**
FROM Playlist, Song
WHERE PlayList.SongID = Song.SongID AND DatePlayed = #26/05/2016#;

16 **2**

department	Lowest Paid
Admin	11.50
Finance	19.50

17a

Membership	Number of Memberships
Full	3
Gym Only	2
Swim Only	1

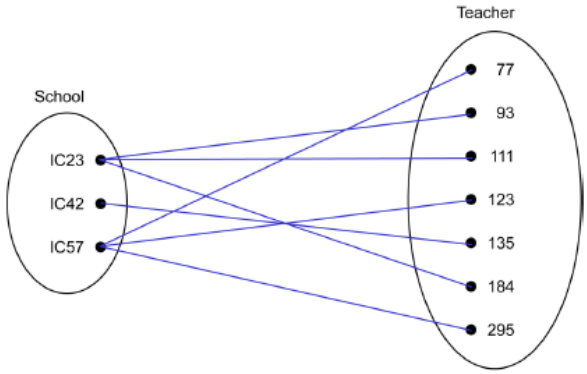
3

17b So that each type of membership only appears once. **1**

18 Entity occurrence diagram to show the following: **2**

1 Manger to 1 department
1 Department to Many Employees

19 **1**



20 Group By should be By Location **2**
COUNT(venueName) should be COUNT(venueName) AS[Number of Restaurants]

21 Relationship between school and teacher entity incorrect – 1 School Many Teachers **3**
SchoolID should be in teacher entity as foreign key
staffID should be foreign key in School entity
staffID should be primary key in Teacher entity

22a **3**

Fields(s) and calculation(s)	Destination
Tables(s)	Bookings
Search Criteria	Departure Date LIKE _ _/10/_ _ _ _
Grouping	
Sort Order	

22b **3**

Fields(s) and calculation(s)	Destination, Number of Bookings = Count(*)
Tables(s)	Bookings
Search Criteria	

Grouping	Destination
Sort Order	

23 SELECT [Pupil ID], [Test Score 1] + [Test Score 2] + [Test Score 3] + [Test Score 4] AS [Total Test Score]
FROM Assessment
ORDER BY [Test Score 1] + [Test Score 2] + [Test Score 3] + [Test Score 4] Asc; **3**