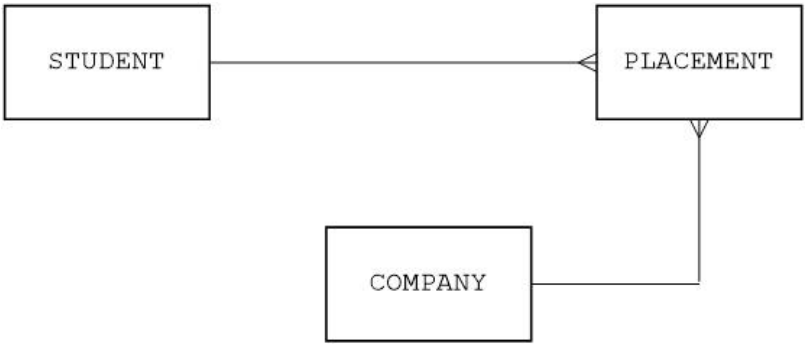
2(a)	**1 mark** for each correct relationship, **max 2** marks 
2(b)	**1 mark** per bullet point, **max 2 marks** • There are no repeating groups of attributes • There are no many-to-many relationships • There are no partial key dependencies // no non-key dependencies // no transitive dependencies

2(c)(i)	**1 mark** per bullet point, **max 2 marks**
	• DELETE FROM and correct table
	• Correct condition
	Example answer:
	`DELETE FROM PLACEMENT`
	`WHERE Complete = TRUE;`

2(c)(ii)	**1 mark** per bullet point, **max 4 marks**
	• SELECT COUNT any appropriate field
	• AS and an appropriate name
	• FROM and correct table and WHERE and one correct condition
	• Two AND clauses with the other two correct conditions
	Example answer:
	`SELECT COUNT (CompanyID)`
	`AS TotalPlacements`
	`FROM PLACEMENT`
	`WHERE CompanyID = "NEAM"`
	`AND StudentID = "LDEA01"`
	`AND Complete = TRUE;`

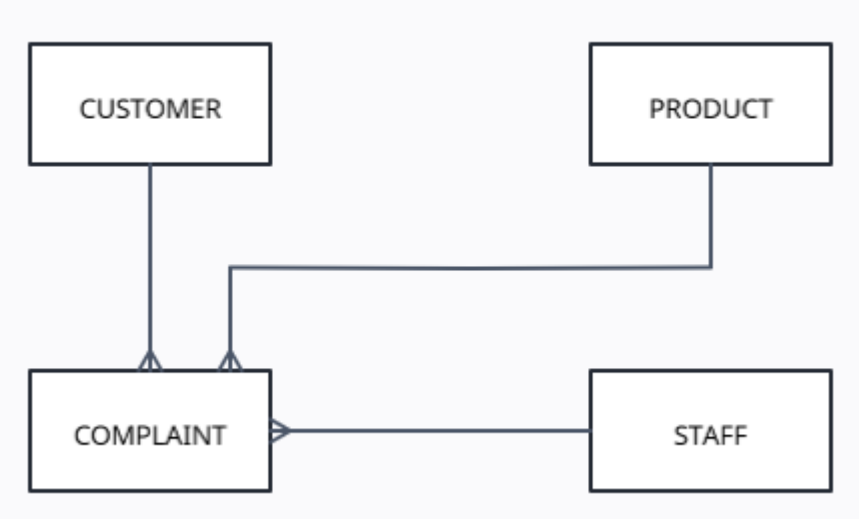
2(d)	**1 mark** per bullet point, **max 3 marks** Generic mark points **(max 2)** • Referential integrity ensures that related data is consistent • Referential integrity ensures that every foreign key has a corresponding primary key • Referential integrity provides for cascading update / delete • Referential integrity ensures that if a primary key is deleted or modified all linked records in foreign table will be deleted or modified // Referential integrity stops "orphaned records"—records in a table that point to an entry in another table that no longer exists Specific mark points **(max 2)** for good examples e.g. • CompanyID is a foreign key in PLACEMENT table and is dependent on the primary key CompanyID in COMPANY table • If a record is deleted from the STUDENT table, then all records with the same StudentID will be deleted from the PLACEMENT table • If a CompanyID is modified in the company table, all records with that CompanyID in the placement table will also be modified

4(a)	**1 mark** per bullet point, **max 2** marks • The relationship between SHIP and CONTAINER is one-to-many (1:M) • The primary key ShipID in the SHIP table is linked to the foreign key ShipID in the CONTAINER table

4(b)	**1 mark** per bullet point, **max 2** marks
	
	• ALTER TABLE statement
	• Including the additional field with a suitable field name and suitable field type
	
	Example answer
	`ALTER TABLE CONTAINER`
	`ADD InspectionDate DATE;`

4(c)	**1 mark** per bullet point, **max 4** marks
	
	• SELECT COUNT statement
	• Using the correct tables
	• Joining the tables
	• The condition for the name
	
	Example Answer 1:
	
	`SELECT COUNT(ContainerID)`
	`FROM CONTAINER, SHIP`
	`WHERE CONTAINER.ShipID = SHIP.ShipID`
	`AND ShipName = "Caledonia";`
	
	Example Answer 2:
	
	`SELECT COUNT(ContainerID)`
	`FROM CONTAINER INNER JOIN SHIP`
	`ON CONTAINER.ShipID = SHIP.ShipID`
	`WHERE ShipName = "Caledonia";`

4(d)	**1 mark** per bullet point, **max 2** marks • To allow the user to create / modify / delete tables // maintain the database • To allow the user to set up / modify relationships • To allow the user to create a form for data input • To allow a user to add tools to a form // for example, drop-down boxes / buttons etc. • To allow a user to design a report (to show the output in an organised manner) • To allow a user to add a menu to enable a choice of different actions • To allow a user to inspect the database contents and metadata • ... by creating and/or running different SQL queries

5(a)	**1 mark** for each correct relationship, **max 3** marks 

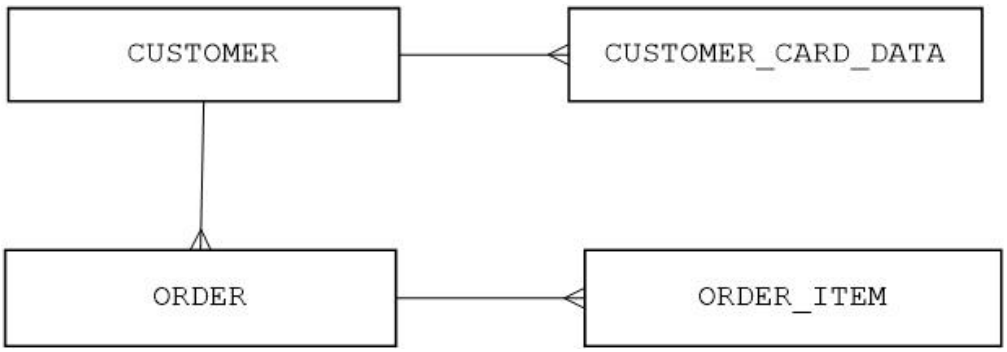
5(b)	**1 mark** per bullet point, **max 4** marks
	
	• Create table with opening and closing brackets and commas separating the attributes
	• Appropriate data types for StaffFirstName, StaffLastName and Department
	• Appropriate data types for StaffID and RemoteWorker
	• Primary key correctly defined
	
	Example Answer 1:
	`CREATE TABLE STAFF(`
	`StaffID INTEGER,`
	`StaffFirstName VARCHAR,`
	`StaffLastName VARCHAR,`
	`Department CHAR,`
	`RemoteWorker BOOLEAN,`
	`PRIMARY KEY (StaffID)`
	`);`
	
	Example Answer 2:
	`CREATE TABLE STAFF(`
	`StaffID INTEGER NOT NULL PRIMARY KEY,`
	`StaffFirstName VARCHAR,`
	`StaffLastName VARCHAR,`
	`Department CHAR,`
	`RemoteWorker BOOLEAN`
	`);`

5(c)	**1 mark** per bullet point, **max 5** marks
	
	• SELECT and the correct attributes
	• FROM and the correct tables
	• Tables joined correctly
	• Correct condition for the rating
	• Correct ORDER BY clause
	
	Example Answer 1
	`SELECT PRODUCT.ProductID, ProductName, ComplaintDetails`
	`FROM PRODUCT, COMPLAINT`
	`WHERE PRODUCT.ProductID = COMPLAINT.ProductID`
	`AND Rating <=5`
	`ORDER BY Rating DESC;`
	
	Example Answer 2
	`SELECT PRODUCT.ProductID, ProductName, ComplaintDetails`
	`FROM PRODUCT INNER JOIN COMPLAINT`
	`ON PRODUCT.ProductID = COMPLAINT.ProductID`
	`WHERE Rating <=5`
	`ORDER BY Rating DESC;`

5(d)	**One mark** for identification of method **one mark** for corresponding description **max 4** marks • Authentication methods / passwords / biometrics / 2-factor authentication can be implemented • ... which prevents unauthorised access to the customer's data • Access rights / privileges can be set • ... so that only those with correct permissions can read / edit customer's data • Regular backups can be scheduled • ... so that a second copy of the customer's data is available in case of loss/damage • The data can be encrypted • ... so that the customer's data cannot be understood by anyone who gains unauthorized access • Different views can be created • ... so that not everyone can see the customer's data

5(a)

1 mark for each relationship. **Max 2** if any extra



5(b)

1 mark for:

CardNumber

5(c)

1 mark per row to **max 2**

Table	Foreign key
ORDER_ITEM	OrderID
ORDER	CustomerID
CUSTOMER_CARD_DATA	CustomerID

5(d)

1 mark each to **max 3**

e.g.

- Each order would only be able to have one item
- or the database would not be normalised
- it would not be in 1NF
- due to repeated groups of attributes
- in the ORDER table

5(e)

1 mark each

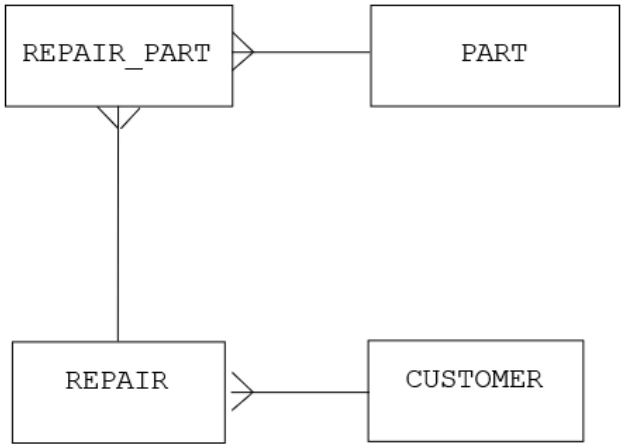
- Selecting the customer ID, customer name and sum of TotalCost with appropriate identifier
- FROM clause with suitable join of tables (ON or WHERE)
- ORDER.Paid = FALSE condition (with correct key word)
- GROUP BY condition

e.g.

```
SELECT CUSTOMER.CustomerID, CUSTOMER.Name,
Sum(ORDER.TotalCost) AS TotalOwed
FROM CUSTOMER INNER JOIN ORDER
ON CUSTOMER.CustomerID = ORDER.CustomerID
WHERE ORDER.Paid = FALSE
GROUP BY CUSTOMER.CustomerID;
```

Alternative JOIN Statement

```
SELECT CUSTOMER.CustomerID, CUSTOMER.Name,
Sum(ORDER.TotalCost) AS TotalOwed
FROM CUSTOMER,ORDER
WHERE CUSTOMER.CustomerID = ORDER.CustomerID
AND ORDER.Paid = FALSE
GROUP BY CUSTOMER.CustomerID;
```

2(a)	**1 mark** for limitation and **2 marks** for corresponding explanation e.g. Data redundancy // data duplication • Separate linked tables are used • Data items are stored once so reduces data duplication / data redundancy Data inconsistency / Poor data integrity • Data changed once in one place will automatically update elsewhere • Linked data cannot be entered differently in two tables • Referential integrity can be enforced The data structure used depends on the application • Changes to the data structure are managed by the DBMS • … and queries are not dependent on the structure of the data • … and changes to the data do not require programs to be re-written
2(b)(i)	**1 mark** for each correct relation 

2(b)(ii)	**1 mark** for each bullet point
	
	• Create table `REPAIR_PART` with opening and closing brackets, all statements inside the brackets
	• `PartID` and `RepairNumber` as varchar (or equivalent)
	• `Quantity` as integer
	• At least 1 appropriate constraint
	• Dual primary key
	
	For example:
	`CREATE TABLE REPAIR_PART(`
	`  PartID VARCHAR(20) NOT NULL,`
	`  RepairNumber VARCHAR(4) NOT NULL,`
	`  Quantity INT NOT NULL,`
	`  PRIMARY KEY (PartID, RepairNumber)`
	`);`

<table>
<tr>
<td>2(b)(iii)</td>
<td>
1 mark for each bullet point

• Select SUM of <code>AmountDue</code>
• From the correct table and one correct condition
• Second correct condition

<pre>SELECT SUM(AmountDue)
FROM INVOICE
WHERE SupplierID = "JK675" AND Paid = FALSE;</pre>
</td>
</tr>
<tr>
<td>2(c)</td>
<td>
1 mark for each correct definition

<table>
<tr><th>Database Term</th><th>Definition</th></tr>
<tr><td>Referential Integrity</td><td>All duplicate entries of data between tables are consistent // all foreign keys are matched to an appropriate primary key</td></tr>
<tr><td>Candidate Key</td><td>A field that could be a primary key but is not // an attribute or smallest set of attributes in a table where no tuple has the same value</td></tr>
<tr><td>Tuple</td><td>A row / record in a table // one instance of an entity in a table</td></tr>
</table>
</td>
</tr>
</table>

6(a)	**1 mark** each to **max 3** e.g. • There is reduced data redundancy // less repeated data • … because each item of data is only stored once • Data consistency is maintained // Data integrity is improved • … changes in one table will automatically update in another • ... linked data cannot be entered differently in two tables • Program-data independence is ensured • … changes to the data do not require programs to be re-written // queries are not dependent on the structure of the data • Complex queries are easier to run • Different views can be provided • …. so users can only see specific aspects of the database • Multiple concurrent access is possible • ... through record locking
6(b)(i)	**1 mark** each to **max 6** • `CUSTOMER` to `JOB` is 1 to many • … implemented by Primary Key in `CUSTOMER` is Foreign Key in `JOB` • `EMPLOYEE` to `LOGIN_DATA` is 1 to 1 • … implemented by Primary Key in `EMPLOYEE` is Foreign Key in `LOGIN_DATA` • `JOB` to `JOB_EMPLOYEE` is 1 to many • … implemented by Primary Key in `JOB` is Foreign Key in `JOB_EMPLOYEE` • `EMPLOYEE` to `JOB_EMPLOYEE` is 1 to many • … implemented by Primary Key in `EMPLOYEE` is Foreign Key in `JOB_EMPLOYEE`

6(b)(ii)	**1 mark** each
	
	• Select SUM of Amount
	• From the correct table **and** one correct condition
	• Remaining correct condition
	
	Example:
	
	`SELECT SUM(Amount)`
	`FROM INVOICE`
	`WHERE Paid = "Y"`
	`AND DateSent >= #01/01/2023# AND DateSent <= #31/12/2023#`

6(a)	**1 mark** each:
	
	• User table with the username as the Primary Key
	• … containing at least email address, date of birth / age and rating
	
	• Quiz table with Quiz ID or date or file name as the Primary Key.
	• … containing at least the other field(s) not used as the PK
	
	• A joining table with an appropriate name including at least fields for user identification, quiz identification and score
	• … with an appropriate Primary Key
	• ... and Foreign Keys matching the Primary Keys of the other two tables
	
	`USER(Username, Email, DateOfBirth, Rating)`
	`QUIZ(QuizID, Date, Filename)`
	`USER_QUIZ(Username, QuizID, Score)`

6(b)	**1 mark** each to **max 2** for data dictionary and **max 2** for logical schema: Data dictionary: • Data about the data in the database // metadata • Identifies the characteristics of the data that will be stored • Appropriate example e.g. field names, table name, validation rules, data types, primary / foreign keys, relationships etc. Logical schema: • Conceptual design • Platform/database independent overview of the database • Is used to design the physical structure • Appropriate example e.g. Design of entities / E-R diagram / views

6(c)(i)	**1 mark** for each correct clause:
	
	• Alter table EVENT
	• Adding foreign key as PlayerID referencing correct table
	
	`ALTER TABLE EVENT`
	`ADD FOREIGN KEY(PlayerID)REFERENCES PLAYER(PlayerID);`

6(c)(ii)	**1 mark** each:
	
	• Selecting PlayerID from EVENT
	• Counting EventID
	• Grouping by the PlayerID
	
	Example:
	`SELECT PlayerID, COUNT(EventID)`
	`FROM EVENT`
	`GROUP BY PlayerID;`

4(a)	**1 mark** for: 1-to-many

4(b)	**1 mark** each:
	
	• Creating table EXAM with opening and closing brackets
	• All fields with appropriate data types and commas at end of lines
	• ExamID as primary key
	
	Example:
	`CREATE TABLE EXAM(`
	`ExamID varchar NOT NULL,`
	`Subject varchar,`
	`Level int,`
	`TotalMarks int,`
	`PRIMARY KEY(ExamID)`
	`);`

4(c)	**1 mark** each:
	
	• Altering table EXAM_QUESTION
	• Linking ExamID to ExamID in EXAM
	
	Example.
	`ALTER TABLE EXAM_QUESTION`
	`ADD FOREIGN KEY (ExamID) REFERENCES EXAM(ExamID);`

4(d)	**1 mark** each to **max 5**: • STUDENT table identified with suitable Primary Key • A linking table between STUDENT and EXAM with suitable Primary Key **and** appropriate name • … that includes the Primary Key of the STUDENT table as a Foreign Key to join with STUDENT • ... and includes the Primary Key of the EXAM table as a Foreign Key to join with EXAM • A linking table between STUDENT and EXAM_QUESTION with suitable Primary Key **and** appropriate name • … that includes the Primary Key of Table 2 as a Foreign Key to join with Table 2 • … that stores the ExamQuestionID and the mark for that question