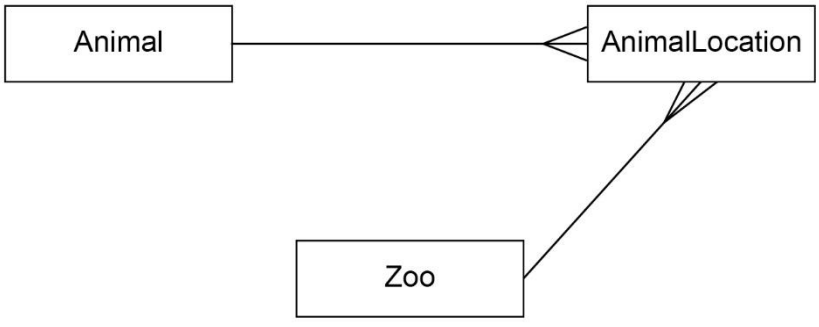
| | | | |
|---|---|---|---|
| 1 | 1 | **All marks AO2 (analysis)**<br><br>Customer(<u>CustomerID</u>, Forename, Surname, TelephoneNumber)<br><br>Appointment(<u>PetID, Date, Time,</u> SurgeryName)<br><br>PetOwner(<u>CustomerID, PetID</u>)<br><br>**1 mark:** Customer relation created and contains the correct attributes and CustomerID identified as the entity identifier.<br>**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.<br><br>**1 mark:** Appointment relation created and contains the correct attributes.<br>**I.** any additional reasonable attributes including CustomerID, VetID<br>**A.** date and Time given as one combined field.<br><br>**1 mark:** Composite entity identifier of PetID, Date and Time identified for Appointment relation.<br>**A.** creation of a new attribute to be the entity identifier eg AppointmentID.<br>**A.** date and Time given as one combined field.<br><br>**1 mark:** PetOwner relation created and contains the correct attributes and no others.<br>Additionally, either:<br>• both attributes identified as a composite entity identifier, or,<br>• **A.** a new entity identifier eg OwnershipID created.<br>**R.** Just one of PetID or CustomerID given as entity identifier.<br><br>**For all mark points**<br><br>**A.** alternative names for relations and attributes created by candidate, so long as meaning is clear.<br>**R.** use of incorrect attribute names for attributes already named in question paper.<br>**A.** spaces in relation and attribute names.<br>**I.** if any unnecessary relations are created.<br>**I.** any representation for foreign keys.<br><br>**Accept responses written in SQL – ignore syntactical errors and data type errors in such responses.** | **4** |

| Question | | | Marks |
|---|---|---|---|
| 2 | 1 | **Mark is AO1 (understanding)**<br><br>**C** The primary key in each relation consists of only one attribute;<br><br>**R.** if more than one lozenge shaded | **1** |

| Question | | | Marks |
|---|---|---|---|
| 2 | 2 | **All marks AO2 (analyse)**<br><br>  <br><br><br>**1 mark**: one-to-many relationship between Animal and AnimalLocation<br>**1 mark:** one-to-many relationship between Zoo and AnimalLocation<br><br>If no marks awarded then award **1 mark** for many-to-many relationship between Animal and Zoo<br><br>**Max 1** if any incorrect relationships drawn (ignore the inclusion of a many-to-many relationship between Animal and Zoo) | **2** |

| Question | | | Marks |
|---|---|---|---|
| 2 | 3 | **All marks AO3 (programming)**<br><br>`AnimalID INT PRIMARY KEY,`<br>//<br>`AnimalID INT,`<br>`PRIMARY KEY(AnimalID),`<br><br>`IndividualName VARCHAR(50),`<br>`Species VARCHAR(40),`<br>`DateOfBirth DATE,`<br>`Sex VARCHAR(6)`<br><br>These **are AO3 marks so syntax must be correct** (including commas) to award them<br><br>**1 mark:** `AnimalID`, with sensible data type and identified as primary key<br><br>**1 mark:** two other fields with sensible data types and lengths (if given)<br><br>**1 mark:** two other fields with sensible data types and lengths (if given)<br><br>**DPT.** data type before fieldname (**Note: penalisation is of marks not mistakes**)<br>**DPT.** incorrect punctuation - missing commas, unnecessary semi-colons, brackets etc but ignore bracket or semi-colon added at very end<br>**A.** Any sensible types. Lengths do not need to be specified<br>**I.** Case<br><br>Valid alternative SQL types are:<br><br>• Alternative types for `AnimalID`: tinyint, smallint, mediumint, integer, number, byte<br>• Alternative types for `IndividualName`, `Species` and `Sex`: char, nchar, nvarchar, ntext, longvarchar, varchar2, nvarchar2, text, tinytext, mediumtext, longtext, string<br>• Alternative types for `DateOfBirth`: datetime, datetime2, datetimeoffset, smalldatetime **R.** time | **3** |

| Question | | | Marks |
|---|---|---|---|
| 2 | 4 | **5 marks for AO2 (analyse) and 2 marks for AO3 (programming)**<br><br>Mark Scheme<br><br>**AO2 (analyse) – 5 marks:**<br><br>**1 mark** for correctly analysing the data model and identifying the tables that data needs to be extracted from (`Animal`, `AnimalLocation`) and the fields that need to be extracted (`IndividualName`, `DateArrived`), and including these and no other tables or fields in the query<br>**A.** inclusion of unnecessary table `Zoo` as long as it is correctly linked to the `AnimalLocation` table by a linking condition<br><br>**1 mark** for correctly identifying the condition to select the correct species of animal: `Species = "Red Panda"` or correctly identifying the condition to select the correct zoo: `ZooName = "Ashdale Park"`<br><br>**1 mark** for correctly identifying the condition to link the two tables: `Animal.AnimalID = AnimalLocation.AnimalID` - see example 3 for how to apply this to nested solutions.<br>**R.** do not award mark if additional linking conditions for tables that the query does not use are included<br><br>**1 mark** for at least one pair of conditions that would identify some animals that were at the zoo during the required period, or **2 marks** for conditions that would identify all animals that were at the zoo during the period. Example conditions (not the only ones) that would do this are:<br><br>Example full set of conditions 1 – award 2 marks for all conditions or 1 mark for any pair of conditions that would identify some animals at the zoo<br><br>`DateArrived < "01/04/2020" AND DateLeft > "31/05/2020"`<br>(animal arrived before and left after time period)<br><br>`DateArrived <= "31/05/2020" AND DateLeft = "01/01/0001"`<br>(animal arrived before end of time period and has not left)<br><br>`DateArrived >= "01/04/2020" AND DateArrived <= "31/05/2020"`<br>(animal arrived during the time period)<br><br>`DateLeft >= "01/04/2020" AND DateLeft <= "31/05/2020"`<br>(animal left during the time period)<br><br>Example full set of conditions 2 – award 1 mark for the DateArrived condition and either of the DateLeft conditions or 2 marks for all three conditions<br><br>`DateArrived <= "31/05/2020" AND (DateLeft >= "01/04/2020" OR DateLeft = "01/01/0001")`<br>(animal arrived before end of time period and left after start of time period or has not left) | **7** |

Example incomplete conditions – award 1 mark for pair of conditions

`DateArrived >= "01/04/2020" AND DateLeft <= "31/05/2020"`
(animal arrived and left during the time period)

**Note:** Award a **maximum of 2 of the 4 available marks** for the correct conditions if they are not joined by the correct logical operators.

**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.

**A.** mark(s) can be awarded for the correct logical conditions even if the required tables are not identified as being used by the query
**A.** > instead of >= and < instead of <=
**A.** ≥, ≤, => and =<

**AO3 (programming) – 2 marks:**

**1 mark** for fully correct SQL in two of the three clauses (SELECT, FROM, WHERE)

**OR**

**2 marks** for fully correct SQL in all three clauses (SELECT, FROM, WHERE)

**Note:**
- For the SELECT and FROM SQL clauses to count as correct SQL, they must have the correct field and table names in them.
- For the WHERE clause to count as correct it must include at least one correct condition, but does not have to include them all (ignore missing conditions or irrelevant conditions), however the whole WHERE clause must have correct syntax.

**A.** table names before fieldnames separated by a full stop
**A.** use of `Alias`/`AS` command eg `FROM AnimalLocation AS AL` then use of `AL` as the table name and note that command `AS` is not required eg `FROM AnimalLocation AL`
**A.** `INNER JOIN` written as one word ie `INNERJOIN`
**A.** insertion of spaces into fieldnames
**I.** unnecessary brackets so long as they would not stop the query working
**A.** use of any type of quotation marks, hashes or no delimiters around dates and times
**A.** > instead of >= and < instead of <=
**R.** ≥, ≤, => and =<
**I.** inclusion of an ORDER BY clause
**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

**Overall Max 6 if solution does not work fully**

Example Solutions

**Example 1 – All conditions in WHERE clause**

```
SELECT IndividualName, DateArrived
FROM Animal, AnimalLocation
WHERE Species = "Red Panda"
  AND ZooName = "Ashdale Park"
  AND Animal.AnimalID = AnimalLocation.AnimalID
  AND
  ( DateArrived < "01/04/2020" AND DateLeft > "31/05/2020"
    OR DateArrived <= "31/05/2020" AND DateLeft = "01/01/0001"
    OR DateArrived >= "01/04/2020" AND DateArrived <=
       "31/05/2020"
    OR DateLeft >= "01/04/2020" AND DateLeft <= "31/05/2020" )
```

**Example 2 – Use of INNER JOIN**

```
SELECT IndividualName, DateArrived
FROM Animal INNER JOIN AnimalLocation ON
  Animal.AnimalID = AnimalLocation.AnimalID
WHERE Species = "Red Panda"
  AND ZooName = "Ashdale Park"
  AND
  ( DateArrived < "01/04/2020" AND DateLeft > "31/05/2020"
    OR DateArrived <= "31/05/2020" AND DateLeft = "01/01/0001"
    OR DateArrived >= "01/04/2020" AND DateArrived <=
       "31/05/2020"
    OR DateLeft >= "01/04/2020" AND DateLeft <= "31/05/2020" )
```

**Example 3 – A Nested Solution**

```
SELECT IndividualName, DateArrived
FROM (SELECT AnimalID, IndividualName
      FROM Animal
      WHERE Species = "Red Panda"
     ) AS RP INNER JOIN AnimalLocation
       ON RP.AnimalID = AnimalLocation.AnimalID
WHERE ZooName = "Ashdale Park"
  AND
  ( DateArrived < "01/04/2020" AND DateLeft > "31/05/2020"
    OR DateArrived <= "31/05/2020" AND DateLeft = "01/01/0001"
    OR DateArrived >= "01/04/2020" AND DateArrived <=
       "31/05/2020"
    OR DateLeft >= "01/04/2020" AND DateLeft <= "31/05/2020" )
```

*Refer nested solutions to team leaders for marking*

| Question | | | Marks |
|---|---|---|---|
| 2 | 5 | **All marks AO2 (analyse)**<br><br>**Advantage (Max 1):**<br><br>It will be quicker to lookup an animal's current location;<br>The current location of an animal can be identified without having to query/search the AnimalLocation relation // only the Animal/one relation needs to be searched to identify the location of an animal // the current location of an animal can be identified with a less complex query/search;<br>**NE.** easier to lookup an animal's current location<br>**R.** it will be possible to identify an animal's current location<br><br>**Disadvantage (Max 1):**<br><br>Additional storage space will be required;<br>This will introduce data redundancy (as the information can already be found from the AnimalLocation relation);<br>Data inconsistency could occur (as the current location in the Animal relation might not match the current location in the AnimalLocation relation);<br>More updates will be required when an animal is moved between zoos;<br>**A.** the database will no longer be normalised | **2** |

| Qu | Pt | Marking guidance | Total marks |
|---|---|---|---|
| 3 | 1 | **All marks AO2 (analysis)**<br><br>Customer(<u>CustomerID</u>, FirstName, LastName, TelephoneNumber)<br><br>Booking(<u>BookingID</u>, ShowingID, CustomerID)<br><br>AssignedSeat(<u>BookingID, SeatNumber</u>)<br><br>**1 mark:** Customer relation created and contains the correct attributes and no others.<br><br>**1 mark:** Booking relation created and contains the correct attributes and no others (but see accept point below).<br>**A.** Inclusion of NumberOfPeople or NumberOfSeats attribute – must be clear from attribute name that this is a count/quantity<br>**I.** BookingID not included (as ShowingID and CustomerID could be composite entity identifier)<br><br>**1 mark:** AssignedSeat relation created and contains the entity identifier from the Booking relation, the SeatNumber and optionally AssignedSeatID.<br>**A.** Entity identifier from Booking relation not identified in Booking relation<br><br>**1 mark** for correct entity identifiers in one or two relations or **2 marks** for correct entity identifiers in all three relations. The correct entity identifiers are:<br><br>• Customer: CustomerID **R.** composite entity identifier of FirstName and LastName, identifier based on TelephoneNumber<br>• Booking: BookingID // composite entity identifier of ShowingID and CustomerID<br>• AssignedSeat: Composite entity identifier of entity identifier from Booking relation and SeatNumber (**A.** including AssignedSeatID) // AssignedSeatID // ShowingID and SeatNumber<br><br>**For all mark points**<br><br>**A.** Spaces in relation and attribute names.<br>**A.** Alternative names for relations and attributes created by candidate, as long as meaning is clear.<br>**R.** Use of incorrect attribute names for attributes already used in relations defined on the question paper.<br>**I.** Any representation for foreign keys.<br><br>**Accept responses written in SQL – ignore syntactical errors and data type errors in such responses.** | 5 |

| Qu | Pt | Marking guidance | Total marks |
|---|---|---|---|
| 3 | 2 | **All marks AO2 (analysis)**<br><br>The `Film` table should not be included // only the `Showing` table should be included;<br><br>The date is missing quotation marks/hashes/delimiters;<br><br>**A.** An asterisk / list of attributes is missing <u>after</u> `DELETE`<br><br>**NE.** Not specified what to delete<br><br>**Max 2** | **2** |

| Qu | Pt | Marking guidance | Total marks |
|---|---|---|---|
| 3 | 3 | **1 mark AO2 (analysis) and 1 mark AO1 (understanding)**<br><br>**1 mark AO2 (analysis):** There might already be bookings for (a showing/showing on) this date;<br>**A.** "There might already be bookings for these showings" without date reference<br>**R.** "There might already be bookings for a/the showing" without date reference<br><br>**1 mark AO1 (understanding):** The database would prevent the query from running as there would be records in the bookings table that referenced showings that no longer existed // if executed the query could leave records/bookings (in the bookings table) that referenced showings that no longer existed // there will be ShowingIDs that reference showings that do not exist;<br>**A.** The foreign key rules might be violated<br>**A.** Referential integrity rules might be violated<br>**A.** Any bookings for the showings would also need to be deleted<br><br>**Award both marks if stated that all bookings for the 29th March would also need to be deleted** | **2** |