



10.5 Client server databases Mark Scheme

Mark schemes

Q1.

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

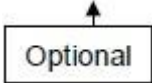
Purge / archive data that is no longer necessary / in use

[12]

Q2.

(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

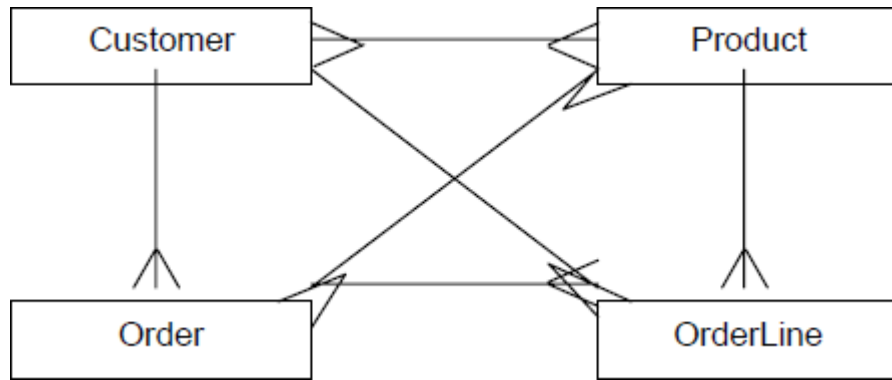
[18]

Q3.

- (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]



Examiner reports

Q1.

A very good range of responses was received to this question, with approximately half of students achieving five or more marks. Most students addressed all three aspects of the question (hardware, network, database and software). Students tended to make more points about how the hardware could be improved than about the other two areas. This was acceptable but students needed to have covered all three areas to achieve a mark of ten or above.

Some students wrote too vaguely to achieve marks, for example by writing that a “faster processor” would improve performance, without referencing a factor such as the clock speed that would make the processor faster. Other mistakes included believing that the question required students to contrast thin-client and thick-client and that the system was web based.

A small number of students wrote about issues which might be causing the system to perform poorly instead of explaining how the performance of the system could be improved. Such responses were not worthy of a mark.

Q2.

Overall, students demonstrated a satisfactory ability to use SQL in question parts (a) to (c).

- (a) This part was the worst tackled of the three. Commonly made mistakes when defining the table in this part were to use incorrect SQL data types, to include the field EngineSize, to declare RegistrationNumber to be an integer or to use non-SQL syntax. Some students attempted to put a constraint on the PolicyType to ensure that it could only be one of the two valid possibilities. Students were not expected to know how to do this, so incorrect attempts at doing so were not penalised.
- (b) In this part, just under half of students got full marks. Many students did not appear to really know the correct syntax of the SQL Update command, but achieved an easy mark by writing the first line of the command as “Update Vehicle”. A common mistake was to fail to put quotation marks around the registration number or colour values.
- (c) Responses to this part were disappointing, given the frequency that students have been asked to write SQL queries in the exam over the years, and the fact that this was a fairly simple example, involving only two tables. Only one third of candidates achieved full marks, though half managed to achieve three of the four marks. As in part (b), one common mistake was to miss quotation marks around the registration number. Other mistakes were to use the keyword GET instead of SELECT and to miss out the AND operator in the WHERE clause. Pleasingly, most students realised that a condition was needed to link the two tables together.
- (d) Parts (i) to (iii) were about server side scripts. Just under half of candidates were able to achieve full marks for part (i) by explaining that a server side script was program code that was executed on a server. Some also recognised that the trigger for this could be the requesting of a web page and that the output would be a web page. Part (ii) was the least well tackled part, with only around one third of students achieving any marks. The Request object is used to fetch user-inputted data from the web server that it will have been sent when the web page that the data was entered on was submitted. The majority of students believed that the Request object

was used to either query a database or that the execution of the command would trigger a request to the user to input data at that time rather than retrieving already input data. Almost all of the candidates got one of the two marks for part (iii) but few clearly explained that the output would be written to a web page which the web server would return to the web browser on the Police Officer's handheld terminal.

- (e) This part was about extending the design of the database to store safety certificate information. The majority of candidates scored at least two of the three marks. Most candidates correctly identified that a new table would need to be created, what the fields in this table would need to be, and that CertificateNumber would be the primary key. Those candidates who scored two but not three marks usually made a mistake with regard to how the new table would be linked to the existing database, stating that the CertificateNumber would be added to the Vehicle table as a foreign key. This solution would not work as this would only allow one certificate to be associated with each vehicle, and it was required that previous certificates could also be stored. The correct solution was to put the RegistrationNumber from the Vehicle table into the new table as a foreign key.

Q3.

Part (a): Most candidates got at least one of the two marks for this question part. Good responses explained both the theoretical aspects of the problem with the order relation and also the real-world consequences of this. Some candidates mistakenly stated that the database contained partial key or non-key dependencies.

Part (b): Most candidates got some marks for this question, but full mark responses were quite rare. A small number of candidates ignored the instruction to draw three relationships and thus limited the maximum mark they were allowed to two out of three. The most common error was to show an incorrect degree for the relationship between the Product and Orderline relations.

Part (c): This question part was very well answered with most candidates getting some marks and many getting all three. The most common mistakes were to define ProductNumber as the primary key, but then to forget to give it a data type, and to declare Price to be an integer data type. When marking this question we allowed the use of data types that were taken from other languages rather than SQL, so long as they were clearly equivalent. In the future it is likely that we will require the use of correct SQL data types and syntax for full marks to be awarded to a response.

Part (d): A server-side script is a sequence of instructions that is executed on a web server to generate web pages dynamically at the time that a request to view a page is received. The majority of candidates showed a reasonable understanding of this and got at least one of the two available marks. Instead of explaining what a server-side script is, some candidates gave examples of usage or described them in such a way that they might have been templates rather than executable programs. A common statement that was insufficient to be creditworthy was that the scripts were stored on a server or accessed from the server. This was not adequate as the same could be said of static HTML pages.

Part (e)(i): This question part was well answered. To achieve both marks, candidates had to make clear that the values were being retrieved from the web server or from the input made in the web browser.

Part (e)(ii): This question part was well answered with many candidates getting both available marks.

Part (e)(iii): Only a minority of candidates achieved the mark for this question part. Most

recognised that a calculation would be performed and the result of this would be output, but to achieve the mark a candidate needed to explain that these results would be displayed on the web page or sent back to the client computer and displayed in the web browser.

Part (f): This question part was well answered. Most candidates clearly understood the structure of an SQL query and many scored high marks. The most common mistakes were to include spurious punctuation such as semicolons or commas in responses and to include the Order relation in the FROM clause, which was not required. The correct command to sort the results was ORDER BY ProductNumber, or alternatively, ORDER BY ProductNumber ASC. Some candidates put brackets around the ASC which is not correct syntax.

