Candidates should be able to:

OCR 2.1.5 (a) describe a database as a persistent organised store of data
AQA 3.1.15 understand the basic concepts of a relational database as a data store

A database is persistent because the data and structures are maintained even when the applications that use the data are no longer running. Databases have to be organised, i.e. structured, so that information can be found and manipulated quickly and easily.

An address book, an encyclopaedia and a telephone directory are examples of paper-based manual databases.

However, it is more common to talk about computerised databases. Electronic databases have several advantages over manual databases. These include:

- the ability for the data to be accessed by more than one person at the same time
- the ability to interrogate or query the data and view the resulting answers
- the ability to for changes to the data to be made quickly available to all end users
- the output of data in a range of different formats to suit user needs (e.g. graphs, reports, forms, etc.) either for viewing on screen or as print-outs

A computerised database is a collection of related data stored in one or more computerised files in a manner that can be accessed by users or computer programs. A supermarket accounting system is an example: as items are sold, the inventory database is updated and the inventory information is made available to the sales staff.

Computers have the ability to store large amounts of data and to process it speedily. Organisations of all sizes use databases to store, sort, interrogate and manage their data. Below are a few examples:

Hospital databases maintain details of patients, doctors and treatments.

Business use databases to keep track of sales, stock and staff etc. and to analyse their own performance.

Internet Search engines, such as Google, Bing, Yahoo etc. all have powerful databases to collect the details of websites that are used in searches.
OCR 2.1.5 (b) explain the use of data handling software to create, maintain and interrogate a database.

Paper databases have a fixed structure. For example, a paper telephone directory is organised by the name of the person or business. As it is not organised by telephone number, it would take an individual a very long time to manually search through for a specific number.

Databases are organised stores so that the data in them can be accessed and processed as efficiently as possible. The data in a database is collected for some specific purpose and a database is usually set up in response to a need.

Directory Services (e.g. 118118) can carry out the same search almost instantly as they use data handling software that is able to rapidly sort and interrogate the data in their telephone directory databases according to their needs.

Modern databases are controlled by a database management system (DBMS). This is software that acts as an intermediary between the data and the applications that the organisation needs. This can range from a simple program that creates and maintains a comma-delimited flat-file database through to sophisticated relational database management systems that can be used to create and manage a huge variety of database structures.
OCR 2.1.5 (c) describe how a DBMS allows the separation of data from applications and why this is desirable

It is undesirable for applications to be data-dependent. The traditional approach to storing data in a company was for everyone to store their own data separately. A company might have a Sales department dealing with customers and a Training department that trained the same customers on the products they'd just bought. These departments would have kept their own data files, with slightly different versions of the data, rather than sharing one file of data.

The problem with this approach is that there are two copies of the same customer details. When a customer moves house they phone the sales person and the address is updated in one place only. The Sales department have the up-to-date details but the rest of company doesn't. This is called “data inconsistency” and is the main problem with duplicating data in more than one place. This is easily solved by everyone sharing one database.

This data duplication leads to wasted storage space and time wasted entering the same information in different places. The ideal situation is to enable applications to share an integrated database containing all the data needed by the applications. This eliminates data redundancy (unnecessary duplication of data).

Ideally, the person responsible for looking after the database, the database administrator (DBA), must have the freedom to change the physical storage of the database structures in response to changing business requirements without having to modify existing applications. When applications are data-dependent, changes made to the database can require major modifications the applications that access it.
OCR 2.1.5 (d) describe the principal features of a DBMS and how they can be used to create customised data handling applications.

The DBMS will manage:

**Security:** data must be kept safe from malicious access

**Creation:** involves using software to define and build the structures to hold the data.

**Maintenance:** involves using software to make changes to the data in the database. For example:

- adding (also referred to as inserting) new data items
- deleting existing data items
- updating (editing) existing data items

**Integrity:** keeping data in such a way that it is consistent, correct, reliable not redundant.

**Querying:** involves using software to search the database for information of interest to the user. E.g. a user of a car showroom database could run a sophisticated query to “find all hatchback or saloon Toyota cars registered between August 2006 and July 2010 but not with 5 doors”.

**Maintaining a log of database access and changes:** many DBMS record details of who accessed the database and when. Most DBMS also record what was changed by that user. This permits the rollback (undoing) of incorrect actions on the database.

**Data validation:** Data validation ensures that, when data is input or changed by users, the DBMS checks the correctness of the changes and either reports invalid data to the user or stores valid data.

**Central control:** A DBMS can provide a DBA with central control of the database. With such control, a DBA can assign access rights and privileges to users and ensure that standards are followed in the representation of data.

**Performance and efficiency:** Knowing the overall requirements of the organisation, a DBA can use the DBMS to structure the database system so that it provides a service that is “best for the business”. Some DBMS provide tools for a DBA to monitor statistics related to database performance and use. If there are certain activities carried out frequently by end users, the DBMS can be adjusted to improve the speed of those interactions. This is particularly important in very large databases.

**Research DBMS**

1. List 4 databases where your details may be stored.
2. List 4 databases a school might use.
3. Microsoft Access is very popular in schools and small businesses but what other Database Management Systems are there? Find another three examples of DBMS.
4. Research how many different software programs could be used with ODBC to connect to an Access database.
OCR 2.1.5 (e) understand the relationship between entities and tables

What is a database entity?

An entity is a “real world thing” about which data is held. Examples of entities include:

- A customer
- A product
- A pupil
- A supplier
- A hotel room
- A DVD
- A flight
- A holiday
- A treatment
- An address book
- A book
- A car
- An order
- An animal
- A student

An attribute is a feature of that entity. For example, a hotel room might have an attribute about whether it has a view or whether it is single or double. A student might have a date of birth and an address.

An entity is stored as a table in a database and an attribute becomes a field in a table. All the data about a particular entity is stored in a single table. Each data item about the entity is a field.
AQA 3.1.15 be able to explain the terms record, field, table, query, primary key, relationship, index and search criteria

What is a database record?
Data in a database table is organised into rows (records) and columns (fields). Each record in a relational database table corresponds to an entity. In the example table of 'Students' above there are 5 records. Each record corresponds to an individual student. Note that although there are two students called Philip Barker with the same date of birth, they have different Student IDs and are different students.

What is a database field?
An attribute is a piece of information or a characteristic of an entity. Attributes of entities are represented in database tables by fields (columns). A field stores one item of data for a record. In the table above, each student is represented in the relational database by a record and the student attributes are stored in the following fields:

- Student ID
- Forename
- Surname
- Date of Birth

Fields have the following characteristics:

- Each field in a table has a unique name. Note, however, that the same field name can occur in other tables of the same relational database.
- Each field stores a single item of data - For example, a field called Date of Birth would store no more than one date of birth value.
- Each field has a particular data type – for example, text, Boolean, integer, date/time, etc.
- Each field can have its own validation rules - these ensure that data recorded in the field is of the right type and format.
Primary Keys
Each table has a primary key. This is a field chosen so that it can uniquely identify each record. Sometimes an existing attribute can be used because it is unique but most of the time some sort of ID is created. Primary keys can be used to link to foreign keys in other tables. A foreign key is the primary key in a different table and it is not necessarily unique.

Relationships
The example below shows how two tables might look with some real data.

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Forename</th>
<th>Surname</th>
<th>Age</th>
<th>Address 1</th>
<th>Address 2</th>
<th>Town</th>
</tr>
</thead>
<tbody>
<tr>
<td>6473</td>
<td>Kit</td>
<td>Kline</td>
<td>37</td>
<td>13 Main Road</td>
<td>Old Town</td>
<td>Hardingshore</td>
</tr>
<tr>
<td>6488</td>
<td>Rose</td>
<td>Jenner</td>
<td>25</td>
<td>148 St. Leonards Lane</td>
<td>The Pringles</td>
<td>Hardingshore</td>
</tr>
<tr>
<td>Test Result ID</td>
<td>Patient ID</td>
<td>Test Performed</td>
<td>Result</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T00011</td>
<td>6473</td>
<td>Pulse</td>
<td>75</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T00012</td>
<td>6473</td>
<td>BP</td>
<td>105/76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T00013</td>
<td>6473</td>
<td>Respiration</td>
<td>45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T00014</td>
<td>6488</td>
<td>Pulse</td>
<td>81</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T00015</td>
<td>6488</td>
<td>Respiration</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In order to create a relationship between the two tables, a link is created using the key field: Patient ID. This means that as each test result is added to the test result table, the only data about the patient that needs to be added is the Patient ID where it becomes known as the foreign key. As can be seen, the Patient ID enables a relationship between the two tables and, apart from the foreign key, there is no duplication of data so no data redundancy. If more details of the patient who was tested are needed they can easily be accessed because of the relationship between the two tables.

What types of relationships can there be in a relational database?

There are three different types of relationship between entities:
- One-to-one – e.g. Husband and wife. A husband can have one wife and a wife can have one husband
- One-to-many – e.g. Political party and politician. A political party has many politicians, but a politician belongs to only one political party
- Many-to-many – e.g. Actor and film. An actor stars in many films and a film can have many actors

Each of the above relationships can be shown in an Entity Relationship diagram (E-R diagram) such as the ones below:
In order to create a relationship between two tables, they must have at least one key field in common i.e. the primary key of one of the tables must be added as a field to the other table where it will is becomes the foreign key.

OCR 2.1.5 (f) understand the components of a relational database, such as tables, forms, queries, reports and modules

**Forms**

Forms are used to create a user interface for data entry. They allow users to type data into a database using a specially designed form rather than typing directly into a database table. The advantage of forms over typing into tables is that forms can be tailored to only accept certain information or only information in the correct format, i.e. forms provide another means of data validation. Drop down list boxes for data selection make the input of data faster and reduce typing errors. In the Contact Details form below, the entry fields for Company, Job Title and Category are selected from a list rather than typed by the user making the data entry.

![A data entry form for Contact Details](image)

Forms can also include graphics such as logos, display image fields, include help text and contain hyperlinks to other forms.

**Reports**

A report is a printed document that contains information from the database that has been organised and presented in a specific way. It is easier to read and obtain the information you need from a report than from looking at the data in the database tables.

Reports are useful for presenting query results in a professional manner. They can also be used to display totals, averages and other calculated fields.
Reports can group, sort and summarise data to give information that is useful for the decision makers in organisations.

For example, a senior sales manager may want to view a report showing monthly product sales by each sales division in order to see which sales teams have been underperforming. See the example on the right:

Most relational databases include facilities for reports to be generated from table data or from the results of queries on data held in the database.

Reports can include charts and formatting to make their information easy to assimilate. As with forms, reports can be customised to include the logos, colours and font styles of an organisation.

What is a query?
The prime function of a relational database is to store data in an organised way so that users can interrogate (search) and manipulate (sort) the data.

The interrogation of a database is called querying the database and a question used to interrogate the data is called a query.

There are many reasons why users may wish to query data, including:

- To identify a group of records that share a certain attribute – e.g. a list of of students with nut allergies, products by supplier, etc
- To calculate totals based on the information held in records – e.g. calculating the total value of the assets held by a company
- To update the details of a specific record or group of records
- To view data in different combinations and formats, etc.

A query design specifies which records the user is searching for and what fields to display out of those records. There are two types of query:

- Simple query
- Complex query
A simple query looks for data in one field only. The example below uses an Microsoft Access database table and a query to find the names of all students who are in form 10B. The Students table must be added to the query, the fields to display are selected and the search criteria (i.e. the specific question) are defined.

Modules
Modules are packages of code that database developers can use to add extra functionality to a relational database. Each module is a collection of procedures that are stored together as a unit in the database. Modules can be associated with specific forms or reports in which case the procedures tend to be specific to the form or report to which they are attached.
OCR 2.1.5 (g) understand the use of logical operators in framing database queries

A complex query looks for data in two or more fields and uses the logical operators OR, AND or NOT.

The following example uses a complex query to find all of the pupils in Form 10B who were born before 1995. This query uses the logical operator AND:

(Form = “10B”) AND (Date of Birth < 01/01/1995).

Operators can be used to refine search results.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>equals</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
</tr>
<tr>
<td>≤</td>
<td>less than or equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>not equal to</td>
</tr>
</tbody>
</table>

The query design is shown below. Note that this time there are two entries in the search criteria row. Also note that this time the query has been given a meaningful name (“Select Query – Pupils in 10B born before 1995”). This saves other database users from unnecessarily creating the same query.

Below is a new complex query that uses the logical operator OR to find pupils who are in Form 10A or Form 10C: (Form = “10A” OR “Form = “10C”) This time, in the query definition there will be two criteria lines. The query and its results are shown below:
What are wildcards in queries?
Wildcard characters can be used in database queries. For example you may want a list of all pupils born in November, or all of the pupils whose surname starts with a ‘C’. Wildcard searches allow you to specify the part of the data that you know and leave the data handling software to fill in the blanks.

Surname Like “C*” would find all records where the surname begins with a C

Sherlock Holmes Whodunnit
http://www.mape.org.uk/activities/whodunnit/

Creating simple queries in Northwind
1. How many orders were shipped to the UK?
2. How many orders were shipped to Colchester, Essex?
3. How many customers live in London? 6
4. What are the contact names of the customer sales representatives who live in London?
   Thomas Hardy, Victoria Ashworth, Elizabeth Brown
5. How many orders were shipped by Speedy Express? 249
6. What are the order IDs of orders for Aniseed Syrup? 12
7. How many customers live in London, or Madrid? 9

Murder Mountain
OCR 2.1.5 (h) explain the use of key fields to connect tables and avoid data redundancy

Your local computer shop wishes to record its sales in a database. Here is a handwritten list of sales produced by the salesman.

<table>
<thead>
<tr>
<th>Order number</th>
<th>Date</th>
<th>Qty</th>
<th>StockId</th>
<th>Description</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>12/07/2008</td>
<td>3</td>
<td>4692</td>
<td>Keyboard</td>
<td>£9.99</td>
</tr>
<tr>
<td>001</td>
<td>12/07/2008</td>
<td>3</td>
<td>5723</td>
<td>Mouse</td>
<td>£4.99</td>
</tr>
<tr>
<td>001</td>
<td>12/07/2008</td>
<td>1</td>
<td>7789</td>
<td>Scanner</td>
<td>£69.90</td>
</tr>
<tr>
<td>002</td>
<td>29/07/2008</td>
<td>1</td>
<td>0134</td>
<td>Laser printer</td>
<td>£124</td>
</tr>
<tr>
<td>003</td>
<td>30/07/2008</td>
<td>1</td>
<td>4692</td>
<td>Keyboard</td>
<td>£9.99</td>
</tr>
<tr>
<td>003</td>
<td>30/07/2008</td>
<td>1</td>
<td>5723</td>
<td>Mouse</td>
<td>£4.99</td>
</tr>
<tr>
<td>003</td>
<td>30/07/2008</td>
<td>1</td>
<td>9834</td>
<td>External Hard Drive</td>
<td>£59.99</td>
</tr>
</tbody>
</table>

It would be more efficient to store the data like this:

<table>
<thead>
<tr>
<th>Order Number</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>12/07/2008</td>
</tr>
<tr>
<td>002</td>
<td>29/07/2008</td>
</tr>
<tr>
<td>003</td>
<td>30/07/2008</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Order Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>Order Number</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>001</td>
</tr>
<tr>
<td>001</td>
</tr>
<tr>
<td>001</td>
</tr>
<tr>
<td>002</td>
</tr>
<tr>
<td>003</td>
</tr>
<tr>
<td>003</td>
</tr>
<tr>
<td>003</td>
</tr>
</tbody>
</table>

The Northwind database is normally supplied with Microsoft Access but can be downloaded:

This database can be used as a case study to examine entities and relationships.

The Northwind Relationships window:
OCR 2.1.5 (i) describe methods of validating data as it is input.

A validation check is a rule that is built into a database to check that the data entered is:

- sensible
- reasonable
- within acceptable boundaries
- complete

It does NOT mean that the data is actually correct, that requires verification.

There are a number of different validation rules that can be used in a database:

**Type Checks** - Field data types provide a basic method of validation. Field data types are assigned to fields during the creation of the database table and data types such as Numeric, Boolean, Date/Time and Image restrict what can be entered. If a user tries to enter text in a date field or alphabetic characters in a numeric field, their entry will be rejected.

**Range checks** - these are used to limit the range of data a user can enter. The 'day' part of a date must be in the range 1 to 31. An exam grade should be in the range 'A'...'G' or 'U'.

**Check digits** - this type of check is used with numbers. An extra 'check digit' is calculated from the numbers to be entered and added to the end. The numbers can then be checked at any stage by re-calculating the check digit from the other numbers and seeing if it matches the one entered. One example where a check digit is used is in the 10 digit ISBN number which uniquely identifies books. The last number of the ISBN is actually the check digit for the other numbers, for example - the ISBN 0192761501.

**Presence checks** - these simply check that an entry has been made in a particular field i.e. a null value (empty field) is not permitted. Usually, not every field in a record needs to be filled in, however there are likely to be some that must have a value and the presence check means that the system will not allow the record to be saved until an entry is made. An application for a passport must have the applicant's surname.

**Length Checks** - All alphanumeric data has a length. A single character has a length of 1 and a string of text such as "Hello World" has a length of 11 (spaces are counted in text strings). A length check ensures that such data is either an exact length or does not exceed a specified number of characters. Mobile phone numbers are stored as text and should be 11 characters in length.

**Lookup** - A lookup check takes the value entered and compares it against a list of values in a separate table. It can then return confirmation of the value entered or a second list based on the value. One use of lookups restricts users to pre-defined input using drop-down lists. A user is forced to use a list box to select from a predetermined list of valid values.

**Input Masks** - Certain alphanumeric fields in a database may require entry in a particular format, e.g. a mixture of numbers and letters. Simple examples of this are dates or text of a specific length. More complex examples include data items such as postcodes, National Insurance numbers, driving licence numbers or product codes. Most databases allow formats to be defined for database fields by an input mask, which defines the valid characters permitted in a field.
Common input mask codes

<table>
<thead>
<tr>
<th>Mask</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>a digit between 0 and 9 must be entered</td>
</tr>
<tr>
<td>#</td>
<td>an entry is optional, but it must be a digit between 0 and 9</td>
</tr>
<tr>
<td>L</td>
<td>a letter between a to z must be entered</td>
</tr>
<tr>
<td>?</td>
<td>an entry is optional, but it must be a letter from a to z</td>
</tr>
<tr>
<td>A</td>
<td>a letter or digit must be entered</td>
</tr>
<tr>
<td>a</td>
<td>an entry is optional, but it must be a letter or digit</td>
</tr>
</tbody>
</table>

**AQA 3.1.15.1** be able to create simple SQL statements to extract, add and edit data stored in databases have experience of using these SQL statements from within their own coded systems

SQL is short for Structured Query Language (pronounced ‘Ess Que Ell) and is a database language that allows users to create, store, update and manipulate data in a database.

**Selecting Data**

Candidates will be expected to be able to use the following SQL commands to retrieve data from tables:

```
SELECT <fieldnames>
FROM <tables>
WHERE <conditions>
ORDER BY <fieldnames>
```

Examples:

```
SELECT *
FROM Cars
WHERE make = "Ford"
ORDER BY hire_date
```

Candidates should be able to use the AND and OR operators together with standard comparisons such as =, >= etc. when specifying conditions in the WHERE clause.

**Inserting, Updating and Deleting Data**

Candidates will be expected to be able to use the SQL commands INSERT INTO, and UPDATE to insert data into tables and update data in tables.

```
INSERT INTO <tablename>
VALUES ( <listofvalues> )
```

```
UPDATE <tablename>
SET <newvalues>
WHERE <conditions>
```

Examples:

UPDATE Cars
SET Colour = ‘Red’
WHERE RegNo = ‘MH09RCM’

SQL reference
An excellent source for SQL reference is:
www.w3schools.com/sql/sql_quickref.asp

Databases
You can download an express version of SQL Server for free at:
www.microsoft.com/express/

You can also download and install MySQL server and the GUI tools for free at:
www.mysql.com

SQL generators
There are some online SQL generators that allow you to type in some SQL to a database that has been described, and see the results:
http://sqlcourse.com/select.html

Microsoft Access
The Northwind database is normally supplied with Microsoft Access but can be downloaded:

This database can be used as a case study to examine entities and relationships and to write QBE and SQL. Assuming that the students have had some time to study and understand the entities and relationships in this database, a good way to start this topic is to suggest increasingly complex queries for students to create using the query designer. Students can then create the same queries in SQL.

The SQL View in Microsoft Access is on the Design View menu when using the query designer.

For example:
1. Who are the customers that live in London?
2. Who are the customers that live in London, or Madrid?
3. What are the contact names of the sales representatives who live in London?
4. What are the order IDs of orders that were shipped by Speedy Express?
5. What are the order IDs of orders for Aniseed Syrup?
6. How many orders were made in September 1997?
7. What is the total value of all orders made on 8 July 1996?

1. SELECT * from customers
   WHERE city = ‘London’
2. SELECT * from customers
   WHERE city = ‘London’
OR city = 'Madrid'

3 SELECT customers.ContactTitle, *
   FROM customers
   WHERE customers.City='London'
   AND customers.ContactTitle="Sales Representative"

4 SELECT Orders.OrderID
   FROM Shippers,Orders
   WHERE Shippers.CompanyName='Speedy Express'
   AND Shippers.ShipperID = Orders.Shipvia

5 SELECT Orders.OrderID
   FROM Products,Orders, [Order Details]
   WHERE Products.ProductName="Aniseed Syrup"
   AND Orders.OrderID = [Order Details].OrderID
   AND Products.ProductID = [Order Details].ProductID

6 SELECT Orders.*, Orders.OrderDate
   FROM Orders
   WHERE Orders.OrderDate>=#9/1/1997#
   AND Orders.OrderDate<=#9/30/1997#

7 SELECT Sum([UnitPrice]*[quantity])
   FROM Orders, [Order Details]
   WHERE Orders.OrderID = [Order Details].OrderID
   AND Orders.OrderDate=#7/8/1996#
   GROUP BY Orders.OrderDate

Do not switch backwards and forwards between SQL View and Design View. Microsoft Access will create inner joins between tables, which may confuse learners.