~Contents~

Part 1: Understanding Databases

1 ~ Why Databases?

A database is a system for collecting, organising and retrieving information; databases are particularly good at working with complex sets of related information.

A database system such as MS Access facilitates a task-driven approach, encouraging you to decide the most effective way to collect, process and present information.

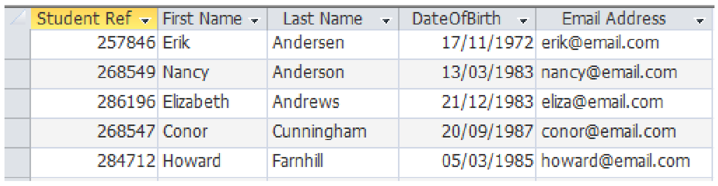
MS Access also includes features to help maintain the accuracy of data by incorporating appropriate checks on validity and data type.

1.1 - Data Structures

Sets of data can be divided into two broad types: **flat-file** and **relational**. The distinction is easiest to explain using an example.

Example 1:

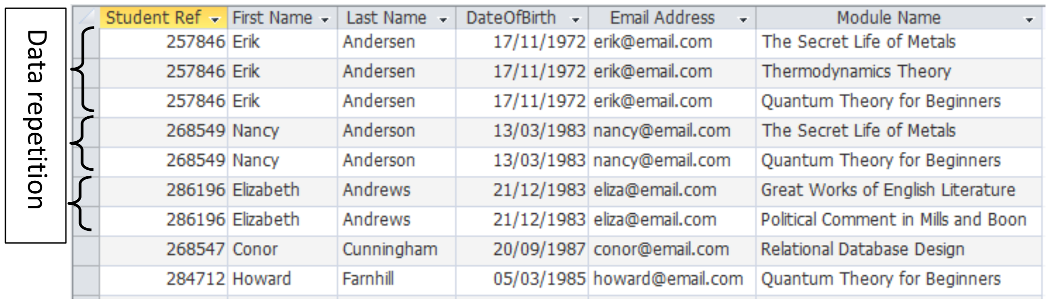
You need to store personal detail for a group of students. A flat-file data structure for this would be a simple two-dimensional table, each student recorded as a row:



Student Ref	First Name	Last Name	DateOfBirth	Email Address
257846	Erik	Andersen	17/11/1972	erik@email.com
268549	Nancy	Anderson	13/03/1983	nancy@email.com
286196	Elizabeth	Andrews	21/12/1983	eliza@email.com
268547	Conor	Cunningham	20/09/1987	conor@email.com
284712	Howard	Farnhill	05/03/1985	howard@email.com

Example 2:

You want to extend this to record which modules are taken by each student, but will need to filter the data set to display details for students taking a particular module. One way to ensure student details will always be visible is to repeat them for each module, but this is a poor solution:



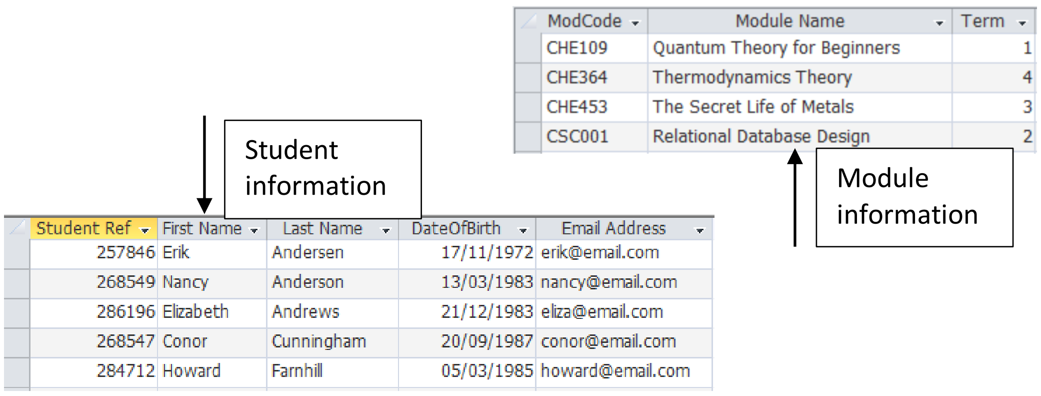
Student Ref	First Name	Last Name	DateOfBirth	Email Address	Module Name
257846	Erik	Andersen	17/11/1972	erik@email.com	The Secret Life of Metals
257846	Erik	Andersen	17/11/1972	erik@email.com	Thermodynamics Theory
257846	Erik	Andersen	17/11/1972	erik@email.com	Quantum Theory for Beginners
268549	Nancy	Anderson	13/03/1983	nancy@email.com	The Secret Life of Metals
268549	Nancy	Anderson	13/03/1983	nancy@email.com	Quantum Theory for Beginners
286196	Elizabeth	Andrews	21/12/1983	eliza@email.com	Great Works of English Literature
286196	Elizabeth	Andrews	21/12/1983	eliza@email.com	Political Comment in Mills and Boon
268547	Conor	Cunningham	20/09/1987	conor@email.com	Relational Database Design
284712	Howard	Farnhill	05/03/1985	howard@email.com	Quantum Theory for Beginners

Disadvantages:

- It provides multiple opportunities to introduce errors

- It takes up more storage space
- It will require multiple records to be changed if one item of personal data changes.

The main problem with this solution, however, is that it does not reflect the *relationship* between students and modules. One student can take several modules, and likewise one module can be taken by several students; the data is **relational** and can never be adequately represented in one two-dimensional table – it requires two:



ModCode	Module Name	Term
CHE109	Quantum Theory for Beginners	1
CHE364	Thermodynamics Theory	4
CHE453	The Secret Life of Metals	3
CSC001	Relational Database Design	2

Student Ref	First Name	Last Name	DateOfBirth	Email Address
257846	Erik	Andersen	17/11/1972	erik@email.com
268549	Nancy	Anderson	13/03/1983	nancy@email.com
286196	Elizabeth	Andrews	21/12/1983	eliza@email.com
268547	Conor	Cunningham	20/09/1987	conor@email.com
284712	Howard	Farnhill	05/03/1985	howard@email.com

MS Excel

Both examples could be implemented using MS Excel, but although the first example is more ‘sensible’ in Excel, it still suffers from disadvantages:

- The same ‘interface’ is used for data input, processing and presentation – programming is required in order to create a data collection form
- Data types cannot easily be enforced (a date could easily be entered as text)
- The integrity of each record cannot be enforced – columns can be re-ordered independently
- Users cannot easily work with a sub-set of the data
- The file cannot be edited simultaneously by multiple users without risks to data integrity

Google Sheets

Most of the disadvantages of Excel also apply to Google Sheets. Even though simultaneous editing is possible, multiple editors do not have genuinely separate views.

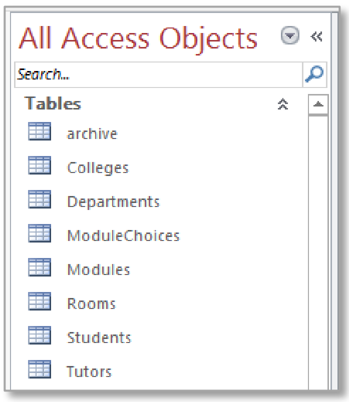
Data can be collected using a Google form, but these cannot also be used to view or present data.

2 ~ Data tables in MS Access

MS Access is designed to facilitate working with relational data. Data are stored in separate tables, but the relationships between these can be clearly defined, enabling you to work with data from multiple tables in a way that reflects their connections.

Tables are one type of 'object' used in Access, with a specific purpose. Other 'objects' serve other purposes and will be used later.

Navigation Panel



All Access objects can be opened for viewing and editing via the configurable navigation pane on the left (this can be minimised to a narrow vertical bar when not in use). To ensure all objects are visible, set to show ✓ **Object type** > ✓ **All Access Objects**

Table Views

An Access table has two views: **Design view** and **Datasheet view**.

To open a table in **Datasheet** View:

- Locate the table in the navigation pane and double-click
  Or locate the table in the navigation pane and choose **Right-click > Open**

To open in **Design** View:

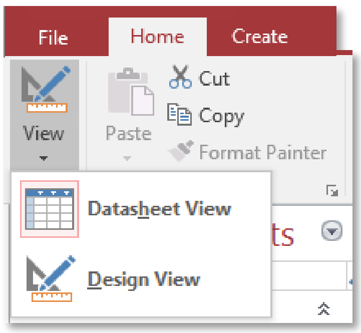
- Locate the table in the navigation pane and choose **Right-click > Design View**

To switch between views when a table is already open:

- Choose **Home > Views > View**

Note: This control is both a toggle control and a drop-down. When working with Access objects you will mostly wish to toggle between Design and Datasheet views so choose the upper portion of the control, not the drop-down.

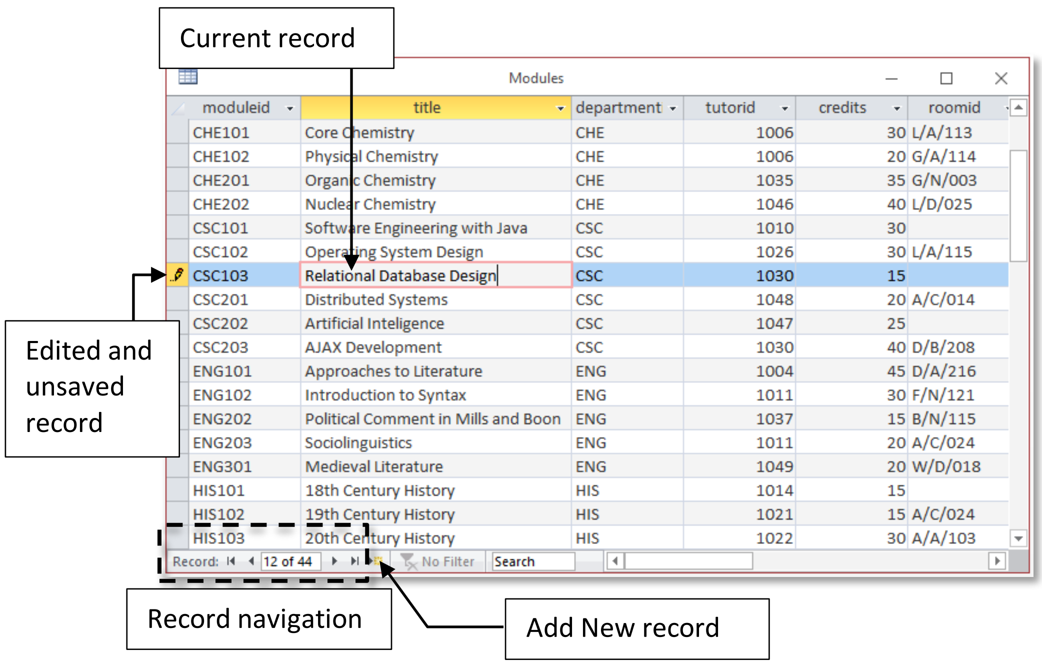
View control →



2.1 - Table datasheet view

The datasheet view presents data in tabular format, where:

- Each column is a **field** of data
- Each row is a **record**
- New records are added in the empty bottom row or using the **New (blank) record** control next to the record navigation controls



moduleid	title	department	tutorid	credits	roomid
CHE101	Core Chemistry	CHE	1006	30	L/A/113
CHE102	Physical Chemistry	CHE	1006	20	G/A/114
CHE201	Organic Chemistry	CHE	1035	35	G/N/003
CHE202	Nuclear Chemistry	CHE	1046	40	L/D/025
CSC101	Software Engineering with Java	CSC	1010	30	
CSC102	Operating System Design	CSC	1026	30	L/A/115
CSC103	Relational Database Design	CSC	1030	15	
CSC201	Distributed Systems	CSC	1048	20	A/C/014
CSC202	Artificial Inteligence	CSC	1047	25	
CSC203	AJAX Development	CSC	1030	40	D/B/208
ENG101	Approaches to Literature	ENG	1004	45	D/A/216
ENG102	Introduction to Syntax	ENG	1011	30	F/N/121
ENG202	Political Comment in Mills and Boon	ENG	1037	15	B/N/115
ENG203	Sociolinguistics	ENG	1011	20	A/C/024
ENG301	Medieval Literature	ENG	1049	20	W/D/018
HIS101	18th Century History	HIS	1014	15	
HIS102	19th Century History	HIS	1021	15	A/C/024
HIS103	20th Century History	HIS	1022	30	A/A/103

Bear in mind, particularly if you are an Excel user:

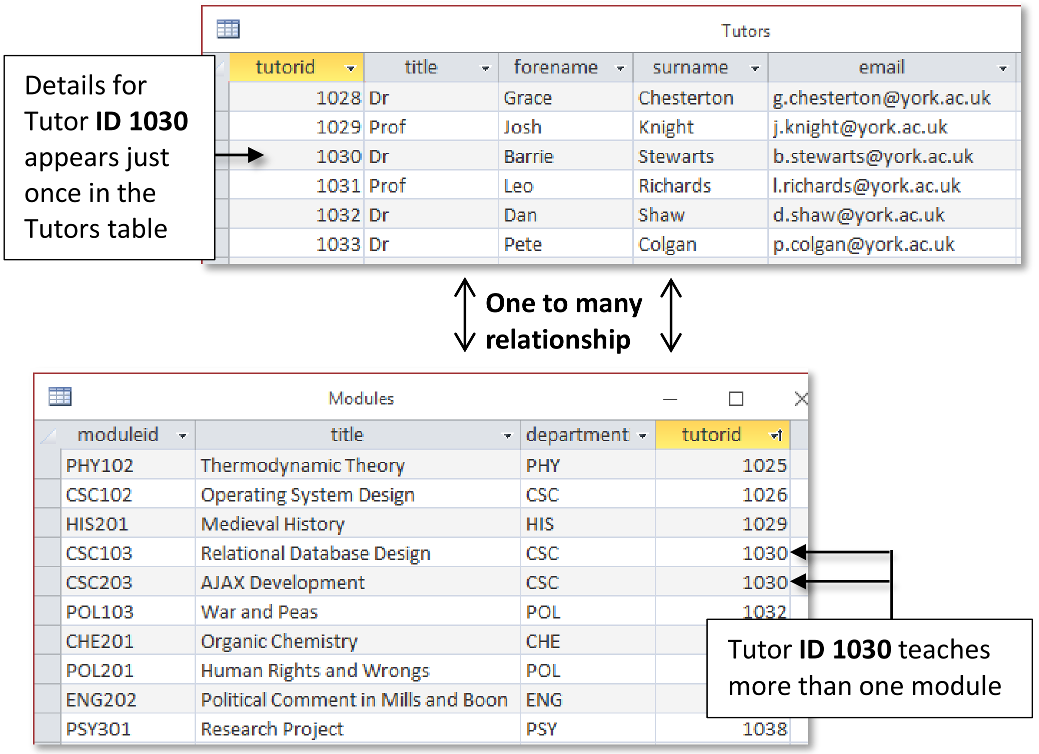
- The integrity of each record (row) is always maintained – you cannot 'shuffle' data in one column independently of others; the record is a key building-block
- The is always only one blank row at the bottom of the table
- When a new record is added, or existing data edited, the unsaved record is indicated by the pencil symbol. Moving to another record will automatically save the edited record. Unlike Excel, ***you do not need to remember to save changes to data***
- The order of records in a table is not important. Later you will use queries to define your view of the data
- You can open and work with several tables (and other Access objects) at once within the main programme window
- Column widths and row heights can be manually adjusted, but all rows will always have the same height

3 ~ Relationships

When we store information about related data in separate tables there must always be a field that links the tables. For example, a tutor could teach multiple modules and so a table containing module information would include the ID of the tutor several times.

3.1 - One-to-many

A separate table would contain the tutor ID along with other information such as their name and email address. Each tutor will appear only once in this table and the Instructor ID will therefore be unique.


Details for Tutor **ID 1030** appears just once in the Tutors table
One to many relationship
Tutor **ID 1030** teaches more than one module

Tutors

tutorid	title	forename	surname	email
1028	Dr	Grace	Chesterton	g.chesterton@york.ac.uk
1029	Prof	Josh	Knight	j.knight@york.ac.uk
1030	Dr	Barrie	Stewarts	b.stewarts@york.ac.uk
1031	Prof	Leo	Richards	l.richards@york.ac.uk
1032	Dr	Dan	Shaw	d.shaw@york.ac.uk
1033	Dr	Pete	Colgan	p.colgan@york.ac.uk

Modules

moduleid	title	department	tutorid
PHY102	Thermodynamic Theory	PHY	1025
CSC102	Operating System Design	CSC	1026
HIS201	Medieval History	HIS	1029
CSC103	Relational Database Design	CSC	1030
CSC203	AJAX Development	CSC	1030
POL103	War and Peas	POL	1032
CHE201	Organic Chemistry	CHE	
POL201	Human Rights and Wrongs	POL	
ENG202	Political Comment in Mills and Boon	ENG	
PSY301	Research Project	PSY	1038

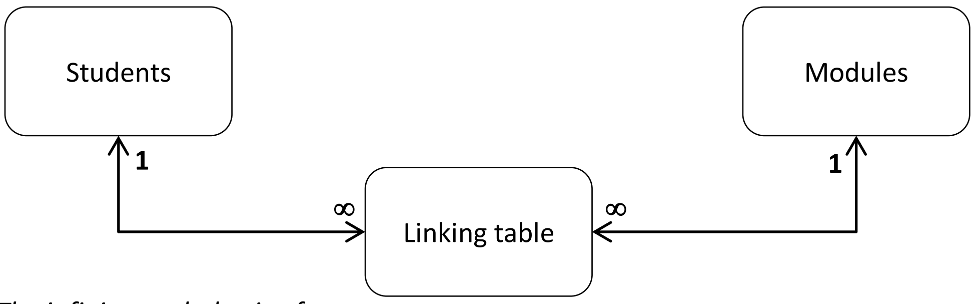
This is known as a **One-to-Many** relationship, as a particular Tutor ID can only exist once in the Tutor table but can appear many times in the Modules table. This is the most common type of relationship.

3.2 - Three-table relationships

This approach would not work for relating students to the modules they were taking. This is because one module could have many students taking it and one student could also take many modules – effectively a 'many to many' relationship between students and modules.

The way round this is to use a 3rd, linking table to record Student IDs and Module IDs. There would be:

- a **one-to-many** relationship between the Student table and the linking table
- a **one-to-many** relationship between the Module table and the linking table



(The infinity symbol ∞ is often used to indicate the 'many' end of the one-to-many relationship)

Two one-to-many relationships

Each record in the linking table would then represent one specific student taking one specific module. Any other information about this instance of a student taking a module could also be included in this table – an exam result, for example.

3.3 - Data Integrity

Given that data are stored in separate tables, it is clearly possible to enter values for Student ID or Module ID in the linking table that have no match in the related Students and Modules tables.

It would also be possible to remove an entry from Students or Modules for which one or more corresponding records exist in the linking table, leaving 'orphaned' records.

Related data in which these errors arise is said to lack **integrity**, and it is important in relational databases that steps are taken to maintain data integrity. This will often be through configuring relationships and will be covered later.

Part 2: Query Essentials

4 ~ Introducing Queries

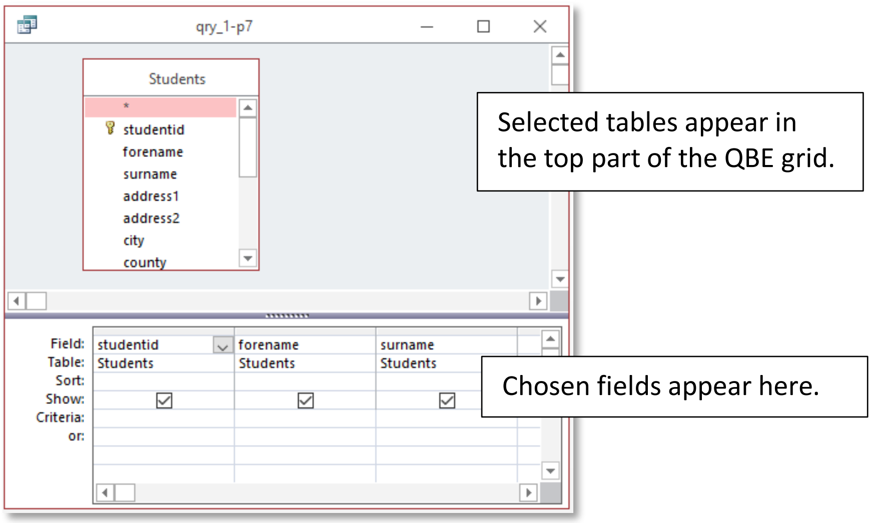
Queries are used for viewing, modifying and deleting records held within database tables. Queries do not themselves store records but contain instructions that describe which records to retrieve from the underlying tables, creating a temporary dataset.

With a query you can:

- Choose which fields of the tables are displayed
- Specify criteria so only the matching records are shown
- Define sorting orders
- Combine data from multiple related tables

4.1 - Constructing queries

Queries are designed and modified using the **Query By Example** (**QBE)** grid, and their results are seen in the **Datasheet** view. You can switch between views using the **Design > Results > View** button (a query must contain at least one data field to be viewable).


qry_1-p7
Students
*
studentid
forename
surname
address1
address2
city
county
Selected tables appear in the top part of the QBE grid.
Chosen fields appear here.

Field:	studentid	forename	surname
Table:	Students	Students	Students
Sort:			
Show:	☑	☑	☑
Criteria:			
or:			

- The top pane of the QBE grid shows any tables used by the query.
- The bottom pane shows the fields from these tables that will be used in the query.

Making a new query

A Query Wizard is included in Access, but it is generally more difficult to use than designing from scratch:

1 Choose **Create > Queries > Query Design** to begin a new query. The **Show Table** dialogue will open automatically.

2 Select the table you wish to use, choose **Add**, then **Close** the dialogue.

3 Add the fields you need to use in the query:

a) **Double-click** on the field name in the table

or

b) **Drag** the field from the table to the lower section of the grid

Editing tables and fields

- Deleting tables: select the table, then press **Delete** (keyboard).
- Adding tables: Choose **Design > Query Setup > Show Table**, select and **Add** the table, and **Close** the dialogue.
- Delete a field: select the field and press **Delete** (keyboard)
- Move a field: select the field and drag it to a new location on the grid

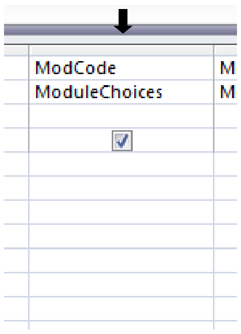
Selecting fields

Deleting and moving fields both require the field on the grid to be selected. To do this:

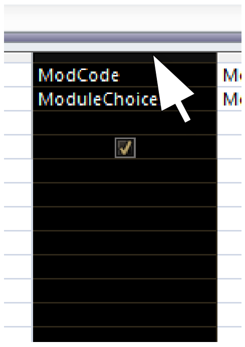
1 Position the mouse pointer at the top of the grid (it changes to a black arrow)

2 Click to select the column (highlighted in black).

Click at the top of the column to select a field



Drag at the top to move it, but position the pointer carefully

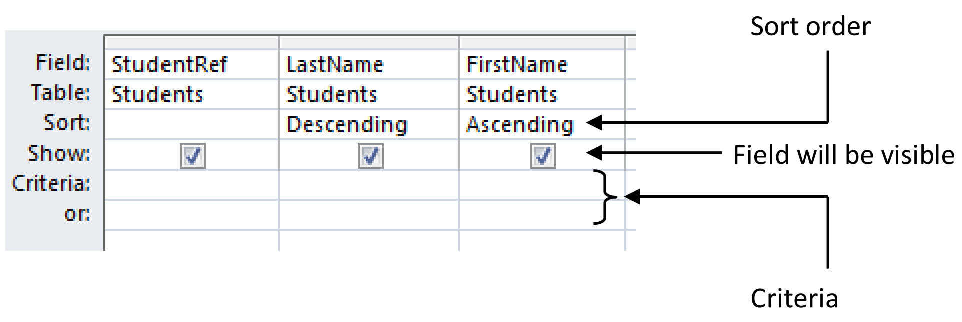


4.2 - Configuring queries

The main things you will want to configure are visibility, sort order and criteria.

Sorting

A sort order can be applied to one or more fields, text and numeric. The sorting is always applied from left to right, so you may need to re-order the fields to get the result you want.



Field:	StudentRef	LastName	FirstName
Table:	Students	Students	Students
Sort:		Descending	Ascending
Show:	✓	✓	✓
Criteria:			
or:			

Visibility

A field will not display in Datasheet view if the box in the **Show** row is not ticked. You are most likely to do this if you want to apply a sort order or criteria to a field but do not want to see that field in the result.

Criteria

Using the criteria row to define which data is retrieved will apply potentially complex filters to your data. It is an essential part of query design, and is covered in detail below.

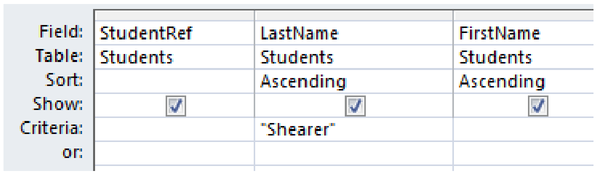
Note: The QBE grid is a visual interface developed for MS Access to construct queries in a language called **Structured Query Language** (**SQL**). Variations of SQL are used by all common database systems.

If you are interested in learning SQL, one approach is to create a simple query using QBE and then choose the SQL View instead of Design or Datasheet. You can then inspect the SQL and figure out the syntax.

4.3 - Filtering in queries

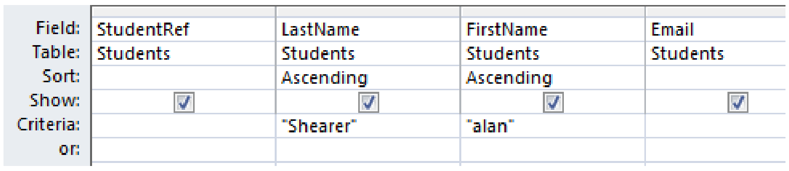
Setting criteria in the QBE grid restricts the results to those records that match the conditions set. Criteria can be set on text, number, and date fields but the syntax is different.

- Criteria for text fields should be enclosed in double quotation marks. They are not case-sensitive.
- Criteria for numeric fields must not be enclosed in quotation marks.
- Criteria for date fields must be enclosed in hash characters.
- Criteria set on more than one field must both be met for a record to be displayed.



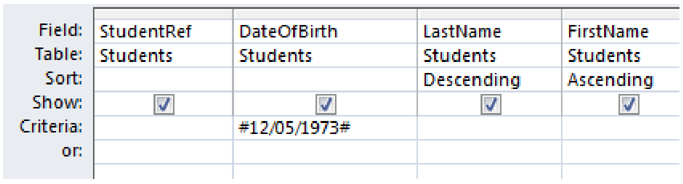
Field:	StudentRef	LastName	FirstName
Table:	Students	Students	Students
Sort:		Ascending	Ascending
Show:	✓	✓	✓
Criteria:		"Shearer"	
or:			

Criteria in text fields



Field:	StudentRef	LastName	FirstName	Email
Table:	Students	Students	Students	Students
Sort:		Ascending	Ascending	
Show:	✓	✓	✓	✓
Criteria:		"Shearer"	"alan"	
or:				

Criteria in a date field



Field:	StudentRef	DateOfBirth	LastName	FirstName
Table:	Students	Students	Students	Students
Sort:			Descending	Ascending
Show:	✓	✓	✓	✓
Criteria:		#12/05/1973#		
or:				

Conditions, ranges and wildcards

A wide range of other symbols and syntax may be used to define criteria more precisely.

Number ranges	>	greater than
	<	less than
	>=	greater or equal to
	<=	less than or equal to
	Between … And …	(insert values)
Date ranges	>	after
	<	before
	>=	on or after
	<=	on or before
	Between … And …	(insert #dates#)
Not equal to	Not *or* <>	
No data ('empty' fields)	A special value of **NULL** is provided for use when locating or excluding records that contain no value in a specific field. A Null value is not the same as a numeric field having a value of zero. **Is Null** – displays only records where no value is entered **Is Not Null** – displays only records where a value is present	

Wildcards for partial matching

Access allows the use of **wildcards** that represent one or more characters when specifying criteria. When using wildcards, the expression must be preceded by the keyword **Like**.

The asterisk symbol * matches 1 or more characters:

- **Like "ch*"** would return any names that begin with Ch such as Charles and Charlotte.
- **Like "*.co.uk"** would return any email addresses that end with .co.uk.
- **Like "*Theory*"** would return 'Quantum Theory for Beginners" and "Thermodynamics Theory".

A question mark ? will match a single character:

- **Like "al?n"** would return 'Alan' and 'Alun' but not 'Allen'

Square brackets [] are used to match a list or range of values:

- **Like "[a,e,i,o,u]*"** returns any value beginning with a vowel.
- **Like "[a-d]*"** returns any value beginning with the letter a,b,c or d.

To exclude a character use the ! symbol:

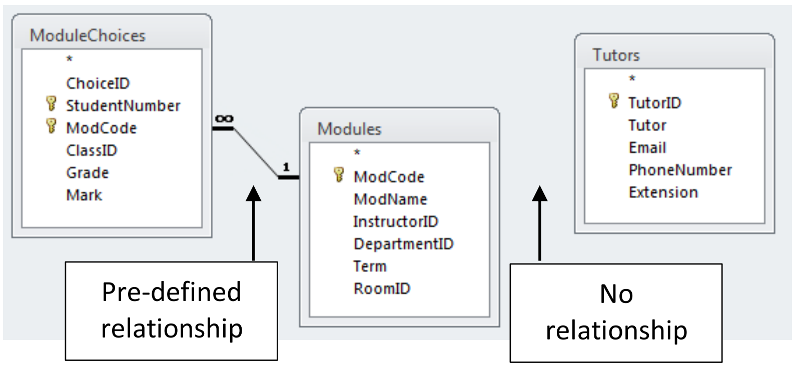
- **Like "[!a]*"** returns all values that do not begin with the letter a.

5 ~ Combining data from related tables

A query can contain data from two or more related tables. When multiple tables are added to the QBE window their relationship must be specified or unexpected results will be returned. There are three possibilities:

- There are already pre-defined relationships between tables (covered in a later section). These will be shown when tables are added to a query.
- A pre-defined relationship may not exist between two tables, but they contain matching field names. A relationship will be created automatically when the tables are added.
- No predefined relationship or obvious corresponding field names exist between two tables. No relationship will be created and you will need to make one.

Whatever the case, the important thing is that any tables you add to the QBE view must be joined correctly, reflecting the relationships between your data.



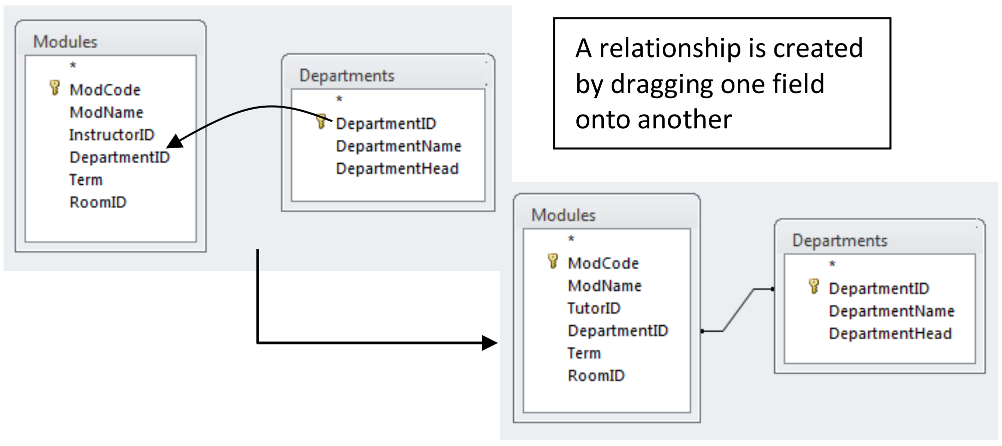
Creating a relationship

1 First identify the two fields (one from each table) that form the connection.

If you cannot identify any common fields, you have probably omitted one or more linking tables.

2 Drag the field from one table onto the corresponding field of the related table.

3 Check the correct fields are used to form the relationship.



Constructing Multi-table queries

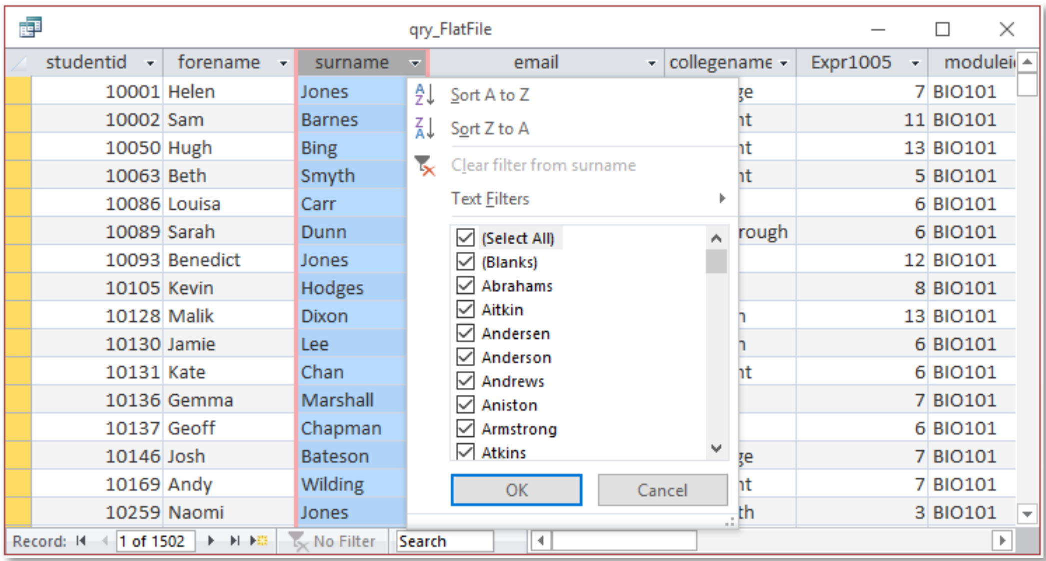
Once you have added more than one table and created relationships, the methods for adding/removing tables, adding/removing fields and configuring the query are essentially the same as for single-table queries.

If you need to add a field that links two tables, it will of course appear in both. Which should you add? With simple queries it will usually make no difference, but may with complex queries; these will be dealt with later.

5.1 - Query quick tools

While it is best to configure sorting and filtering within the query design, it is also possible to apply these to the results of a query in datasheet view.

- In Datasheet view, the heading of each field shows a small arrow. Select this to bring up sort and filter options for that particular field (as in an Excel list).



studentid	forename	surname	email	collegename	Expr1005	modulei
10001	Helen	Jones		ge	7	BIO101
10002	Sam	Barnes		nt	11	BIO101
10050	Hugh	Bing		nt	13	BIO101
10063	Beth	Smyth		nt	5	BIO101
10086	Louisa	Carr			6	BIO101
10089	Sarah	Dunn		rough	6	BIO101
10093	Benedict	Jones			12	BIO101
10105	Kevin	Hodges			8	BIO101
10128	Malik	Dixon		n	13	BIO101
10130	Jamie	Lee		n	6	BIO101
10131	Kate	Chan		nt	6	BIO101
10136	Gemma	Marshall			7	BIO101
10137	Geoff	Chapman			6	BIO101
10146	Josh	Bateson		ge	7	BIO101
10169	Andy	Wilding		nt	7	BIO101
10259	Naomi	Jones		th	3	BIO101

- Sorting and filtering tools can also be found on the **Home** tab.
- On the **Home** tab the **Find** button can be used to locate records within a particular field or the whole dataset.

Note: These controls can be used in the datasheet view of both tables and queries.

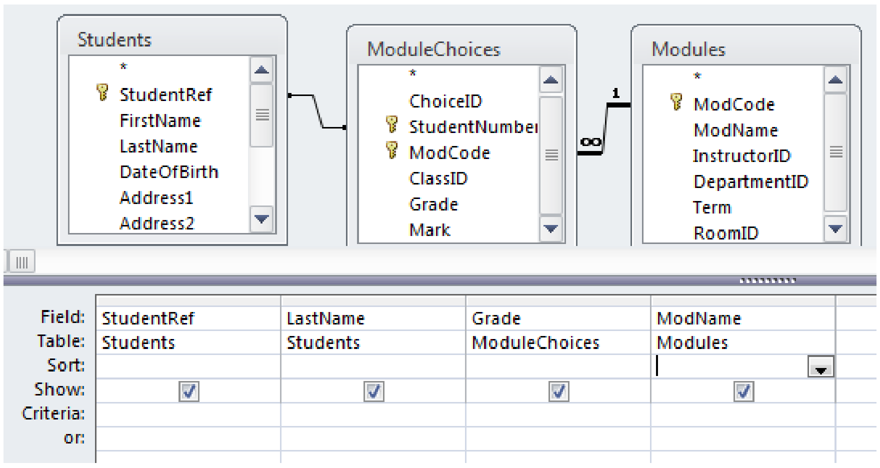
6 ~ Data editing

Queries are designed to allow you to view the data you need for a particular task; this may also include adding or editing data. The records returned in a query can be modified in the same way as in a table, but because queries can display information from multiple related tables, restrict the number of records returned, and only display the desired fields, it may be an advantage to use a query instead.

For a single-table query, this will be straightforward, but when several related tables are used you may find you are only able to edit specific fields.

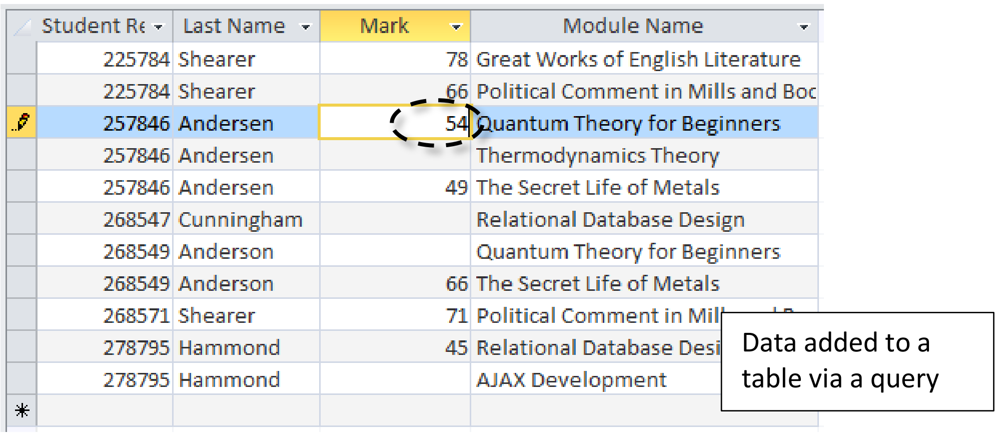
Multi-table example

The query below shows combined information from the Student, ModuleChoices and Modules tables.



Field:	StudentRef	LastName	Grade	ModName
Table:	Students	Students	ModuleChoices	Modules
Sort:				
Show:	✓	✓	✓	✓
Criteria:				
or:				

In datasheet view you can enter marks for each module studied by a student – these will be added to the ModuleChoices table. Depending on the configuration of underlying tables and relationships you may or may not be able to edit student data (eg Last Name) and Module Data (eg Module Name).



Student Re	Last Name	Mark	Module Name
225784	Shearer	78	Great Works of English Literature
225784	Shearer	66	Political Comment in Mills and Boc
257846	Andersen	54	Quantum Theory for Beginners
257846	Andersen		Thermodynamics Theory
257846	Andersen	49	The Secret Life of Metals
268547	Cunningham		Relational Database Design
268549	Anderson		Quantum Theory for Beginners
268549	Anderson	66	The Secret Life of Metals
268571	Shearer	71	Political Comment in Mil
278795	Hammond	45	Relational Database Desi
278795	Hammond		AJAX Development
*			

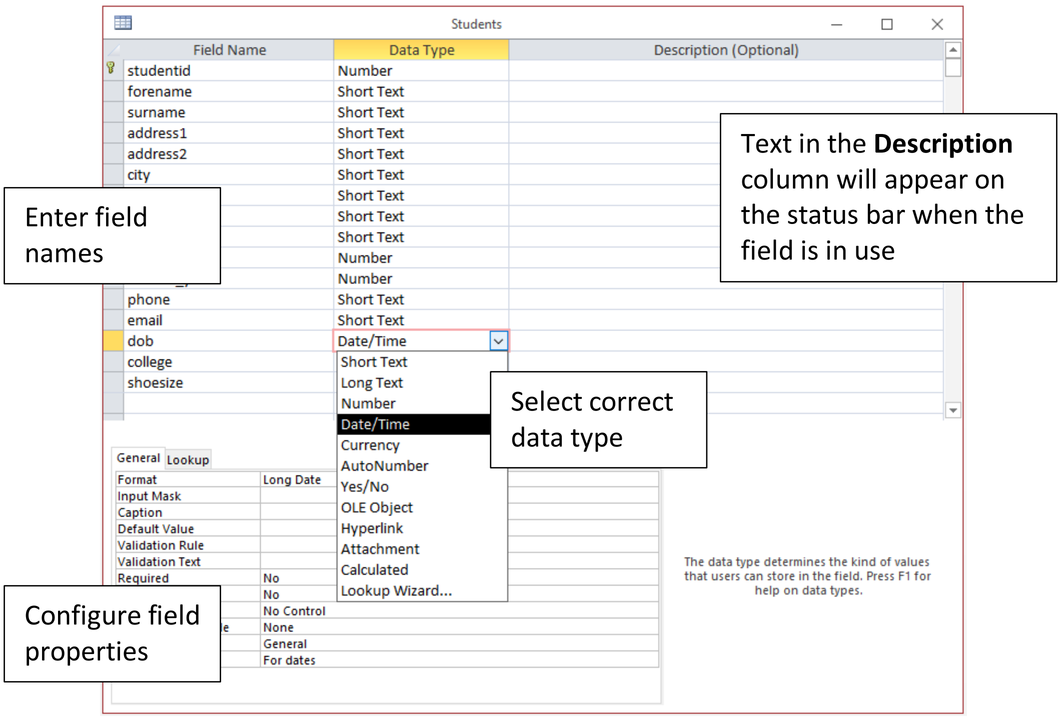
Part 3: Data Tables

Tables store data. All other Access objects work with this data. In a relational database data may be distributed between several related tables.

You need to configure all the tables in which data are stored; we recommend you use **Design** View for this as it gives access to all properties.

7 ~ Configuring Fields

Table Design view lists the fields in the first, **Field Name** column. New fields are added by entering a field name in this column and configuring appropriately.



Field Name	Data Type	Description (Optional)
studentid	Number	
forename	Short Text	
surname	Short Text	
address1	Short Text	
address2	Short Text	
city	Short Text	
	Short Text	
	Short Text	
	Short Text	
	Number	
	Number	
phone	Short Text	
email	Short Text	
dob	Date/Time	
college		
shoesize		

Format	Long Date
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
	No
	No Control
	None
	General
	For dates

The data type determines the kind of values that users can store in the field. Press F1 for help on data types.

Choice of field names: Access is fairly flexible, but you should aim for consistency and clarity, particularly where the 'obvious' field name may be obvious in more than one table – eg Surname.

- As you will sometimes need to type field names, choose reasonably short ones. Each field may also include a more user-friendly caption (see below)
- Access can help with the syntax of typed expressions, but will struggle if names include spaces - use underscore or *camel-case* instead.

Some examples:

Student Surname – bad choice, includes a space

Student_Surname – uses underscore

StudentSurname – camel-case

StudSName – camel-case and abbreviated

As each field is added, it must be configured appropriately. Configuration aims to:

- ensure data are stored in the most appropriate format
- minimise the space required for the database
- help prevent errors in data entry

7.1 - Data Types

As with most databases, you must specify the type of data to be stored in a table field.

Note *Excel users:* whereas Excel will 'guess' the data type when you enter a value in a cell, and is happy to mix them in a column, with Access you must declare the data type of each field when the table is created.

Access is more forgiving than some systems (you can calculate with numbers in a 'text' field), but you should always make the effort to define data types correctly. The range of data types is summarised below:

Data Type	**Use for…**	**Examples**
Short Text	Text and 'numbers' that do not require calculation (maximum 255 characters)	Names, addresses, telephone numbers
Long Text (previously 'Memo')	Longer text (>255 characters)	Making notes
Number*	Any numerical data potentially requiring calculation or sorting	5, -193, 3.002, -17.42
Currency	Money, with an automatically displayed symbol	£199.99 €250
Date/Time**	Dates and times	16/06/1993
Yes/No	Data that has only two possibilities	Yes - No, True – False, 1-0
AutoNumber	Used to assign an automatic unique ID to each records	1, 2, 3, 4, 5, 6…
OLE Object	Photographs, documents	Links to files
Hyperlink	Web and file path links	www.york.ac.uk
Attachment	Used to attach documents	
Calculated	Opens expression builder to calculate a value from other fields	

***** Will also require property setting to define range of values allowed

****** Will also require property setting to define date/time format to be used

7.2 - Field Properties

Some properties are essential to ensure the correct data format will be used: others are features that make designing easier or help reduce errors. To configure properties:

1. Switch to **Design** view if necessary.
2. Select each field in turn.
3. Set **Field Properties** using the panel in the lower section of Design view.
4. Save the table design.

The properties vary with data type and are summarised here:

Text field properties

Field Size	This determines the maximum number of characters that can be stored in the field. Leaving it at the default 255 will 'reserve' that much space for every record, so may make your database file bigger than it needs to be. Set it to a value slightly larger than the longest field entry you anticipate. **Note**: You can change it later if you find it is too large/small.	
Format (optional)	**>**	Force display of text into uppercase
	<	Force display of text into lowercase

Number field properties

Field Size	For numbers this determines the range and precision of values in the field. If you know all values in a field will be integers then choose an integer option.	
	Byte	Positive integers up to 255
	Integer	Integers between -32,768 and +32,767
	Long Integer	Integers from about -2x10^9 to + 2x10^9
	Single	Positive and negative decimal numbers, from about 10^{-45} to 10^{38}, to 7 decimal places
	Decimal	Positive and negative decimals, from about -10^{38} to 10^{38}, to 28 decimal places
	Double	Positive and negative decimals, from about 10^{-300} to 10^{300}, to 15 decimal places
Format	After setting field size, use **Format** to choose other display options. If you want a fixed number of decimal places (rounded values) then select **Fixed** and also set the number of decimal places in the **Decimal Places** property to a value other than *Auto*.	
Decimal Places	For **Field Size** and **Format** combinations that display decimal places, use this property to control the decimal precision.	

Date/Time field properties

Format	Select the required date/time format for display. Access stores dates and times as decimal numbers, so this does not affect the stored values.

Other useful properties

Caption	Use the **Caption** property to add user-friendly display names to fields with unfriendly names.
Default	When a new record is added to a table, any fields with a value set in the **Default** property will automatically contain that value, avoiding the need to enter it. Quotation marks are needed for text, but not for numbers. The value inserted by this method can still be changed if required.
Validation Rule **Validation Text**	These two properties are used together. Data will be checked against the **Rule** and only allowed if it complies. Example rules: **<=50** (less than or equal to 50) **>Date()** (a date after the current date) The **Text** is the message to be displayed if data breaks the rule.
Required	If a field has the **Required** property set to **Yes**, it cannot be left blank when a record is created or edited.

Date and Time values

Access includes a method for using the current date and or time values in an expression. In the context of table properties these can be used when setting default values or validation. The three values are:

Date()	current date
Time()	current time
Now()	current date and time

Using in *Default* property:

- Record the date/time a record is created by entering Date(), Time() or Now() in the default property.
- Automatically enter a date that is a specific number of days before/after today's date.

Using in *Validation*

- Check if a date value entered in a field is reasonable by testing if it is before/after today's date (you can use **>, <** and **=**).

8 ~ Key Fields

Relational databases expect that each record in a table has a unique identity, and the usual way to achieve this is to include a field that will contain a different value for each and every record. This field is then referred to as the **Primary Key**.

8.1 - Primary Key

Access will offer to create a **Primary Key** if you try to save a table without one. Don't automatically agree to this – see if it makes sense to define your own and if no unique field already exists, invent one: a User ID, Customer Reference, Item ID etc.

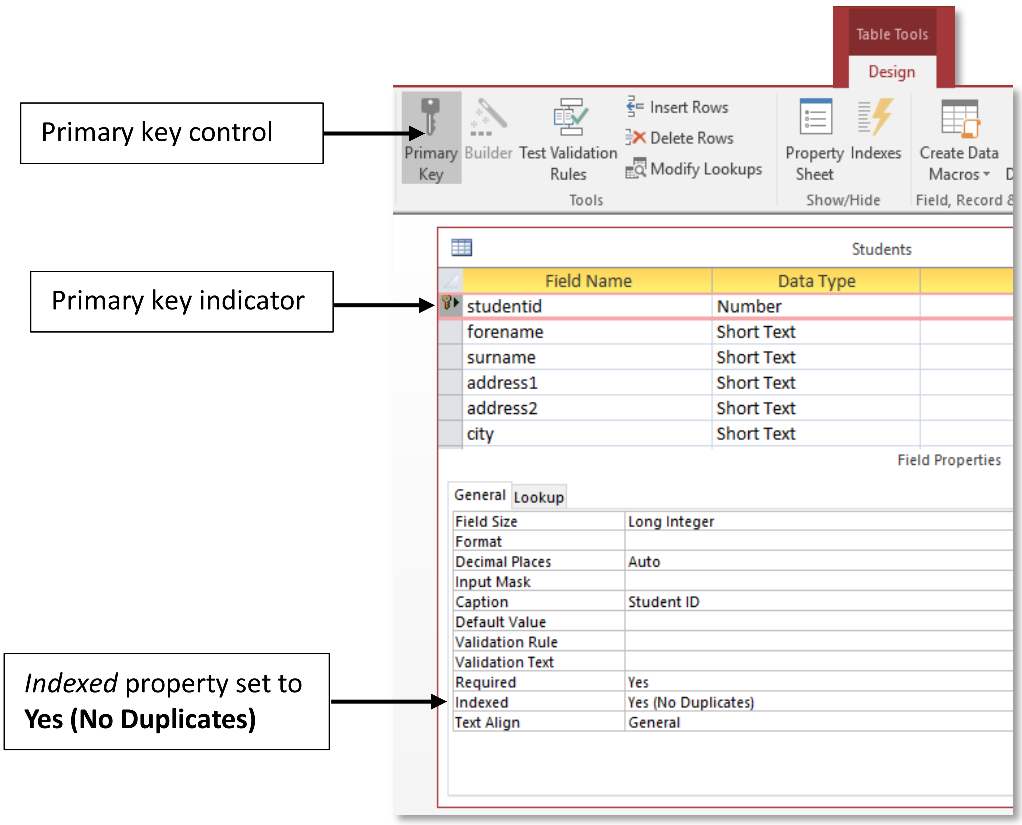
Defining a Primary Key:

1 In table **Design** view, select the field to be used as the primary key

2 On the **Table Tools Design** tab, select **Tools > Primary Key**

The **Indexed** property is affected when the primary key is set:

Indexed	A primary key will have this property set to **Yes (No Duplicates)**. This prevents the same value being entered more than once in records for this field. **Note:** There are other uses for the Indexed property, but usually all other fields have this property set to **No**.



Field Name	Data Type
studentid	Number
forename	Short Text
surname	Short Text
address1	Short Text
address2	Short Text
city	Short Text

Field Size	Long Integer
Format	
Decimal Places	Auto
Input Mask	
Caption	Student ID
Default Value	
Validation Rule	
Validation Text	
Required	Yes
Indexed	Yes (No Duplicates)
Text Align	General

AutoNumber

When there is no obvious attribute to use as a Primary Key, one option is to allow Access to number each new record incrementally. To do this:

1 Create a field with a suitable name and set the data type to **AutoNumber**

2 Configure this field as the **Primary Key**

AutoNumber versus User-Defined Key

Although you could always choose AutoNumber for primary keys, there are advantages in using existing data:

- It may relate to other systems more easily (eg an existing User ID)
- If it is constructed from other data, it enables validation of primary key values (the UK Driver Number is constructed from surname and date of birth)
- Additional information may be conveyed (traditional landline numbers indicate country, town and local area)

8.2 - Foreign Keys

The Primary Key identifies table records uniquely, and it can therefore be used to represent (and so link to) a specific record in that table. This means its value will often be entered in another table several times to represent that item.

The field that stores these values will be called the **Foreign Key** – a Primary key used in another table. In this case:

- the primary and foreign key fields must use the same data type (with one exception)
- the field properties for both must be the same, except for the Indexed property, which should be **Yes(No Duplicates)** for the *Primary Key* and **No** for the *Foreign Key*
- there is no requirement for identical field names but it is recommended and will make constructing queries more straightforward

Property exception

If the Primary Key has the Data Type **AutoNumber**, the corresponding Foreign Key must have the Data Type set to **Number**, with the **Long Integer** Field Size property.

8.3 - Composite Keys

In cases where two or more fields would provide a unique value, these may be used together as the Key (often as an alternative to using an AutoNumber field). For example, an optician could only see one person at a time so an appointments table could use fields such as *Date*, *Time* and *Optician ID* as a **composite key**.

9 ~ Defining relationships

Relationships between tables can be created when a query is designed. More features are available, however, if they are configured separately; in particular you can choose to enforce referential integrity and decide how Access will respond to changes in critical data.

9.1 - Referential Integrity

Relational databases rely on the existence of consistent data in separate tables. If no steps were taken to maintain its integrity, it would be possible to 'break' the database very easily. You could, for example:

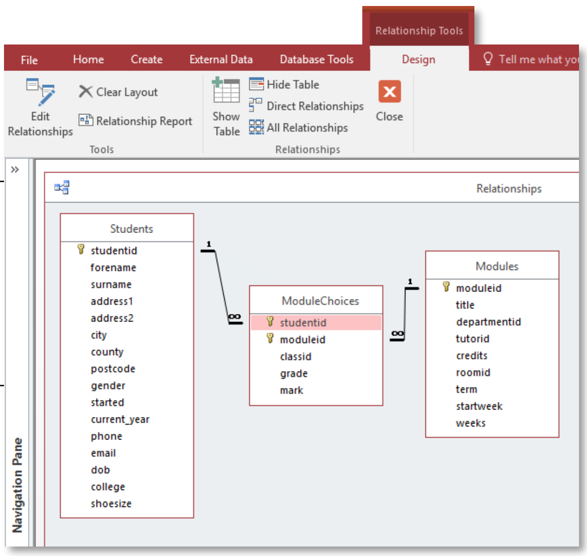
- Enter a non-existent Student ID into another table that uses Student ID as a foreign key
- Change the value in a primary key field (eg Student ID) when the value also exists in other tables as a foreign key
- Delete a Student record, when records for that student exist in other related tables

In order to prevent these occurring, when creating relationships Access enables you to enforce referential integrity, whilst also allowing you to change and delete data when necessary.

9.2 - Creating relationships

Relationships are managed using a diagram that shows tables, including lists of fields, and any joins that have been made.

Relationships are managed using a diagrammatic interface



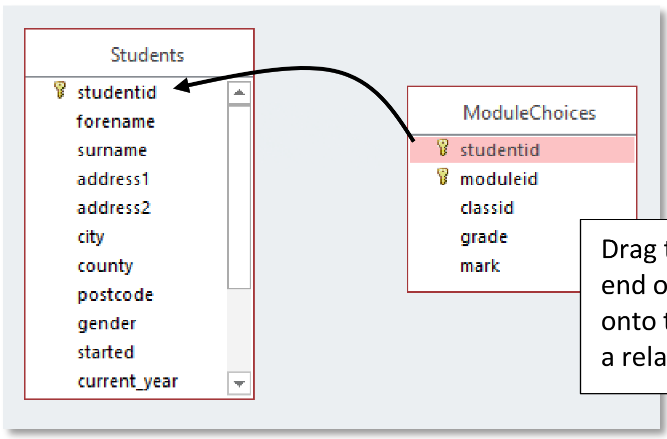
Before defining a relationship, check that the tables contain the appropriate fields – usually a primary key and foreign key.

1 Open the relationships window by choosing **Database Tools > Relationships > Relationships**.

The first time you open this window the **Show Table** dialogue will present a list of available tables. To open this dialogue on future occasions select **Relationship Tools > Design > Relationships > Show Table**.

2 Add the tables for which you want to define a relationship (select and **Add** – use CTRL for multiple selections); **Close** the dialogue when you've added them.

3 Identify the two fields that will make the relationship and then **drag** one onto the other, ensuring you position the pointer accurately.

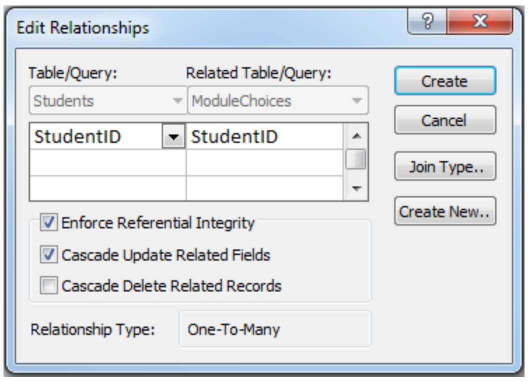


Drag the field for one end of the relationship onto the other to create a relationship

4 The **Edit Relationships** dialogue opens, with the tables and linked fields pre-entered. Choose how you wish to enforce integrity. The options are:

Enforce Referential Integrity	Only allow an existing value for the primary key to be entered into the related table where it is used as a foreign key
Cascade Update Related Fields	If the value in the primary key is changed, make the same change to corresponding values in the related table where it is used as a foreign key
Cascade Delete Related Fields	If a record is deleted from the table containing the primary key, also delete all related records from the related table

Check the correct fields are linked and choose how to enforce integrity


Edit Relationships

Table/Query: Students
Related Table/Query: ModuleChoices
StudentID — StudentID

[✓] Enforce Referential Integrity
[✓] Cascade Update Related Fields
[ ] Cascade Delete Related Records

Relationship Type: One-To-Many

Comments

- Having created these relationships they will be reflected in any new query you create; however, you can change the join type if necessary.
- If you are unlikely to make changes to key field values, or to do so would be problematic, enforce referential integrity without any cascade options.
- Think carefully before choosing Cascade Delete as you cannot undo deleted records.
- By default Access warns before deleting records, but this warning can be disabled by a designer.
- This dialogue also displays the type of relationship detected. If you expect *One-To-Many* but *One-To-One* is shown, this usually means the foreign key field is configured incorrectly and does not allow duplicate values.

Join Type

This dialogue allows you to configure the join type, which determines whether records are displayed from the primary table when no matching records are available in the related table.

The default is to display only matching records (an **inner join**). The alternative (an **outer join**) is more usually configured within a query.

10 ~ External Data

In many practical examples, data to be used in an Access database already exists in another form – an Excel spreadsheet, or an export from another system. Such data does not need to be re-entered as it can be imported.

10.1 - Importing Data

Data can be imported into Access if it is in a recognisable format. Common formats include other Access databases, Excel, CSV and other text formats.

Importing is carried out using a Wizard and although different choices need to be made for different types of data file, the process is similar. When importing data you can create a new table or append data to an existing one. Appending can be problematic as the field order and data types must match exactly.

Access tables

1 Select **External Data > Import & Link > Access**. An import dialogue opens

2 Select **Browse** and locate the Access database file

3 Make sure the **Import** option is selected rather than the **Link** option, then choose **OK**.

4 A full list of all the tables will be presented. Choose one or more to import – use CTRL to select more than one – and select **OK** to import the data.

Excel

When importing from Excel using the Wizard, a view of the data is provided to help ensure you make the correct choices. You have the option to append the data to an existing table, but for this to be successful the field order and data types must match exactly.

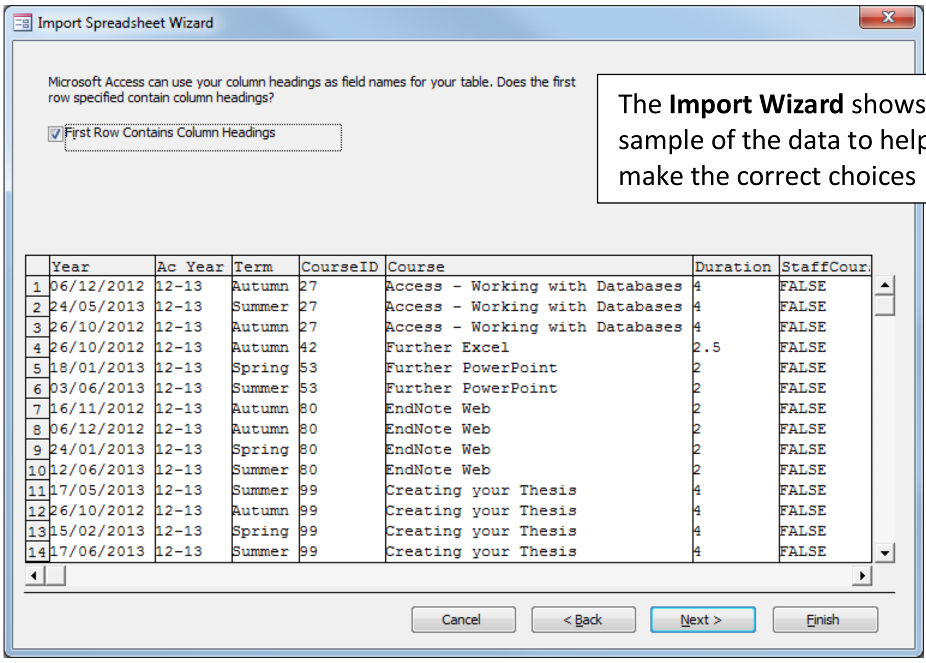
1 Select **External Data > Import & Link > Excel**, choose **Browse** and locate the Excel file.

2 Make sure the **Import** (or Append) option is selected rather than the **Link** option – to Append, you must specify a table – then choose **OK**.

3 The worksheets available are listed – select the one you need and check the data, then choose **Next**.

4 In the next few steps you must make the choices that are appropriate to your data – whether the list has column headings, which field is a primary key, data types, etc. It's usually OK just to accept the defaults. A table name is then suggested, which you can accept or change.

5 Finally, you may opt to save the import settings for future use. This is useful if you regularly import data from the same spreadsheet.



Microsoft Access can use your column headings as field names for your table. Does the first row specified contain column headings?

[✓] First Row Contains Column Headings

	Year	Ac Year	Term	CourseID	Course	Duration	StaffCour
1	06/12/2012	12-13	Autumn	27	Access - Working with Databases	4	FALSE
2	24/05/2013	12-13	Summer	27	Access - Working with Databases	4	FALSE
3	26/10/2012	12-13	Autumn	27	Access - Working with Databases	4	FALSE
4	26/10/2012	12-13	Autumn	42	Further Excel	2.5	FALSE
5	18/01/2013	12-13	Spring	53	Further PowerPoint	2	FALSE
6	03/06/2013	12-13	Summer	53	Further PowerPoint	2	FALSE
7	16/11/2012	12-13	Autumn	80	EndNote Web	2	FALSE
8	06/12/2012	12-13	Autumn	80	EndNote Web	2	FALSE
9	24/01/2013	12-13	Spring	80	EndNote Web	2	FALSE
10	12/06/2013	12-13	Summer	80	EndNote Web	2	FALSE
11	17/05/2013	12-13	Summer	99	Creating your Thesis	4	FALSE
12	26/10/2012	12-13	Autumn	99	Creating your Thesis	4	FALSE
13	15/02/2013	12-13	Spring	99	Creating your Thesis	4	FALSE
14	17/06/2013	12-13	Summer	99	Creating your Thesis	4	FALSE

The **Import Wizard** shows a sample of the data to help you make the correct choices

Other Text Formats

Data from other systems may be imported, but you need to be able to identify how data items are separated. The two main options are:

- **Delimited** – a specific character occurs between fields (comma, tab, space)
- **Fixed width** – each field occupies the same number of characters, with extra spaces used for padding

Examples:

CSV File (delimited with a comma):

"MOD_CODE","MOD_NAME","MOD_CRDT"

"4210100","The Making of the Middle Ages", 20

"4210103","Crusading Europe c.950-1250", 20

TAB-delimited (→ indicates a tab character):

"MOD_CODE" → "MOD_NAME" → "MOD_CRDT"

"4210100" → "The Making of the Middle Ages" → 20

"4210103" → "Crusading Europe c.950-1250" → 20

Fixed-width text file:

```
MOD_CODE····MOD_NAME······················MOD_CRDT

4210100·····The Making of the Middle Ages····20······

4210103·····Crusading Europe c.950-1250······20······
```

Import Method:

1 Select **External Data > Import & Link > Text File**, choose **Browse**, locate the file and **OK**.

2 Make sure the **Import** (or Append) option is selected rather than the **Link** option – to Append, specify a table – then choose **OK**.

3 In the next few steps you must make the choices that are appropriate to your data, checking that it is being correctly separated into fields (columns) in the sample view.

One of the steps will allow you to choose a field to be the primary key.

4 A table name is suggested, which you can accept or change.

5 Finally, you may opt to save the import settings for future use. This is useful if you regularly import data from files of the same structure.

10.2 - Post Import Checks

If errors occur during import (problems identifying the appropriate data type are common) these will be saved in an error table and you may need to do some manual tweaking.

In the table design, always check the following:

Text fields	Check the data type is set to **Text** and adjust the **Field Size** property as an import will often set all text fields to the maximum size of 255 characters. **Note:** If you reduce the field size Access will warn you when Saving changes, but as long as you have counted the number of characters required for a field correctly you won't lose data.
Number fields	Check the data type is **Number** and configure the **Field Size** property for the range and precision you require. **Note:** Any change to number properties will produce warnings on when you Save the change, but as long as the configuration matches your data it should convert correctly.
Primary Key	If you chose a primary key during import this should be set correctly, but always check the key is set and the **Indexed** property is **Yes (No Duplicates)**
Others	Check that: • date/time fields have been identified correctly • yes/no fields have been converted correctly

10.3 - Linked Data

Access can incorporate data from other sources without importing a copy, but by creating a link. Linking to existing data can help reduce duplication.

Characteristics of linked data

- Any changes made to the linked data by other people will be reflected in your Access database without having to re-import
- You may have read-only permissions for data so may not be able to make changes
- You will have no control over the design of the linked data (eg data types, primary key)

Linked data is ideal in situations where you want to make use of data that is managed using another system, application or Access database.

Linking external data

The precise method depends on the type of data source you are linking to, so these are general pointers rather than detailed instructions.

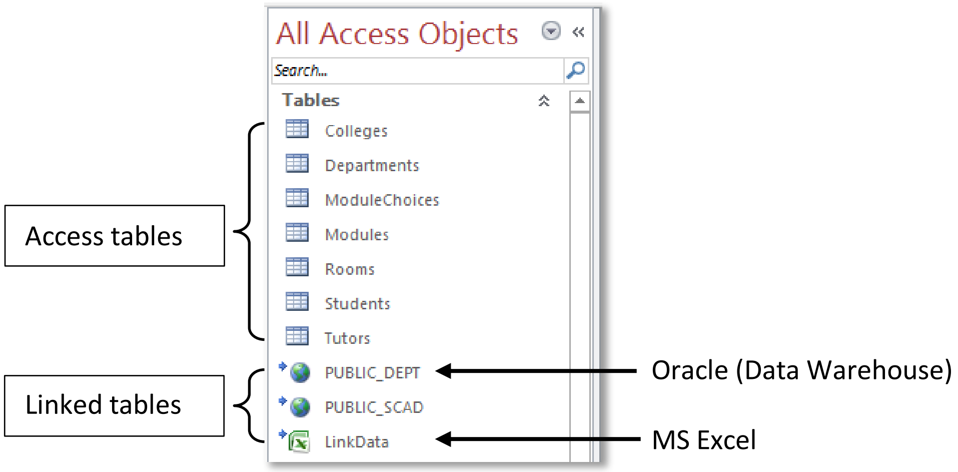
1 First make sure you have access to the location of the data to be linked. You must have at least ‘read’ permission in order to create the link.

2 On the **External Data** tab (**Import & Link** section) choose the appropriate data source (Excel, Access, ODBC etc)

3 Depending on which you choose, you will be presented with an appropriate dialogue box to complete. Make sure you select the **Link** option when presented.

Using Linked Data

Once a table has been linked, the data it contains can be used in queries just like any other data. In the object list linked tables are shown with alternative icons to indicate the presence of a link and the type of source.

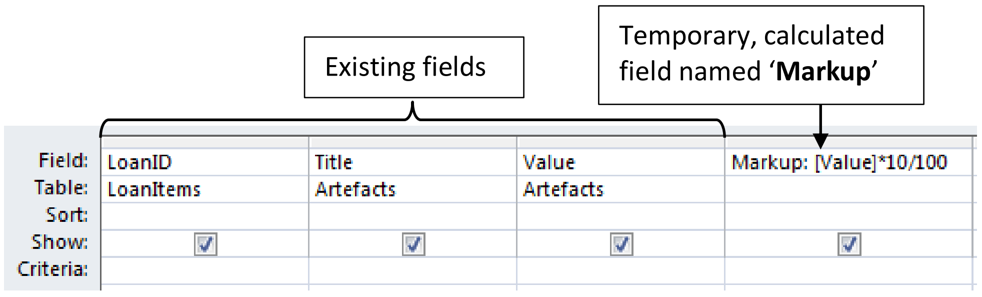


Part 4: Creative Queries

11 ~ Data Manipulation

A query can carry out various operations, including calculations, on the data retrieved from tables and display it in a temporary field. In Access this is done by entering the details in an empty column in the QBE grid:

- The name of the temporary field must be followed by a colon
- Fields used in the expression must be present in the tables in the query and field names must be enclosed in square brackets
- Additional text must be enclosed in double quotes and the ampersand character (&) used to join items
- Further manipulation, including fields can be carried out in reports and forms



Field:	LoanID	Title	Value	Markup: [Value]*10/100
Table:	LoanItems	Artefacts	Artefacts	
Sort:				
Show:	✓	✓	✓	✓
Criteria:				

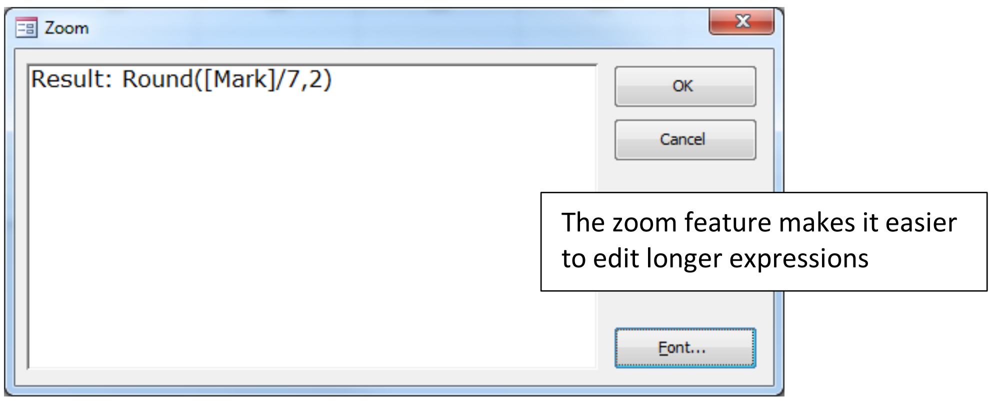
Zoom feature

To help with editing expressions in the **Field** row you can zoom the contents:

- **Right-click** on existing contents in the Field row and choose **Zoom...**

  or whilst editing contents in the Field row use **SHIFT + f2**

The contents are shown in a pop-up dialogue and can be edited here. Use the Font… control to change the display in the zoom control (this doesn't affect the query display).



Result: Round([Mark]/7,2)

11.1 - Calculated fields with numeric data

- The common arithmetic operators can be included (+ - * / ^)

- Brackets can be used to control the order of precedence or avoid ambiguity
- Numbers can be used but must not be enclosed in quotes
- Mathematical functions such as Sqr()or Round() can be used

Examples:

[Cost] * [Quantity] * 10%
[Price] – [Discount]
Round([Mark]/5,2)
Date() + 14

Number format

If the result is of a query will be used to generate a report or form (see later) then you would probably want to decide on number formats at this later stage.

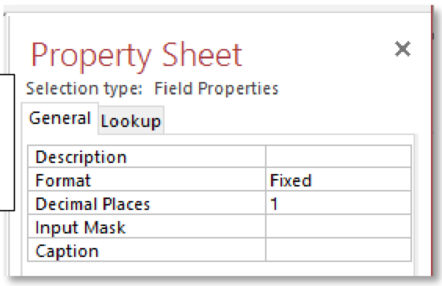
However, if you need to control the number format for the datasheet view of a query, you can do this via the query properties.

1 With the query in design view, select **Query Tools > Design > Show/Hide > Property Sheet** to view the properties sheet.

2 The property sheet will always show properties for the current object – click anywhere in the field to be formatted to show the property values for this field.

3 On the **General** tab, choose the appropriate format from the Format drop-down. If you choose **Fixed** decimal places, also enter the number of decimal places to display. Access will round values correctly when using this option.

The format of individual fields can be configured using the properties sheet



Description	
Format	Fixed
Decimal Places	1
Input Mask	
Caption	

11.2 - Fields with text data

There are also a number of built-in functions that can be applied to data from a text field. These include:

- UCase – converts the text to uppercase
- LCase – converts text to lowercase
- Left – Returns a specified number of characters starting from the Left

  Commonly used to produce an initial from a name: **Left([FirstName],1)**

Concatenation

Any number of fields can be joined together (concatenated) by separating their field names with an ampersand (&). New text enclosed in speech marks can also be joined.

Spaces are not added when joining fields, so you may need to add the space character (enclosed in speech marks) into the formula.

Examples:

```
"Telephone Number: " & [TelNumber]
[FirstName] & " " & [LastName]
Left([FirstName],1) & " " & [LastName]
```

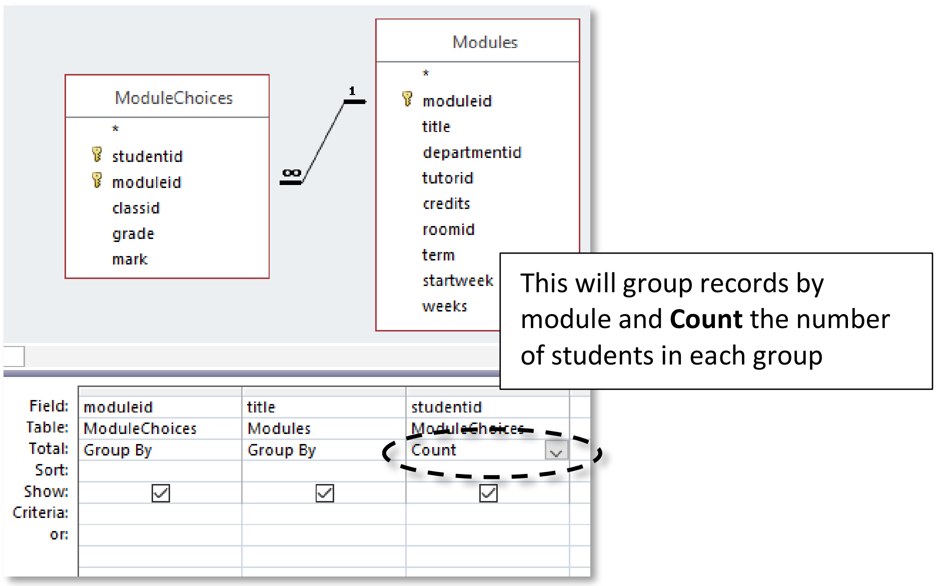
Note: Concatenation can be used with both text and numbers.

11.3 - Grouping and totals

Queries can be used to calculate and count a group of similar records. You could, for example, count the number of students taking each module:

1. Begin a new query in design view.
2. Identify the field containing values you need to count (total, average etc) – in the example shown, StudentNumber – and add to the QBE grid.
3. Determine the field(s) which identify the groups of similar records – in this example we want to count the number of students in each module. Add to the grid.
4. Enable the **Totals** row by selecting **Design > Show/Hide > Totals**.
5. In this new **Total** row, select the required function for the appropriate field.

Tip: Use as few fields as possible, and only apply a function on one field.



This will group records by module and **Count** the number of students in each group

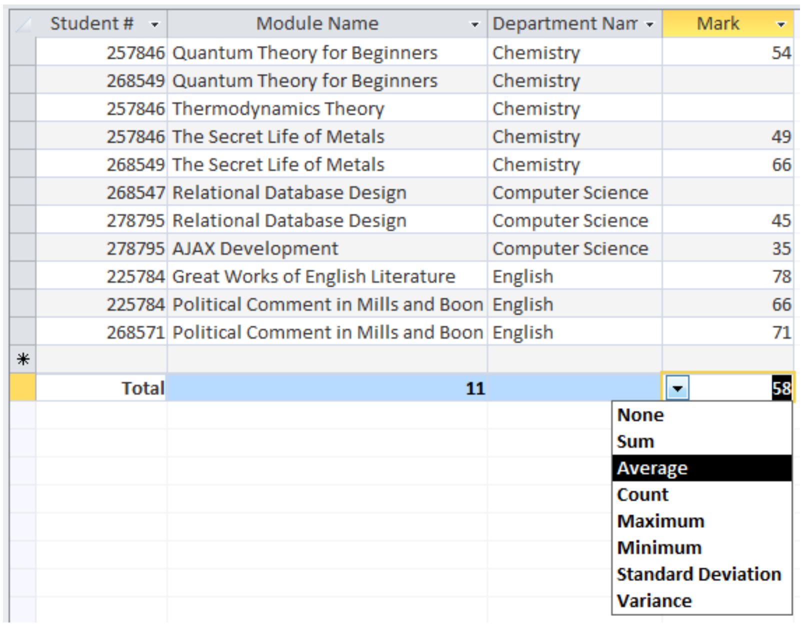
Field:	moduleid	title	studentid
Table:	ModuleChoices	Modules	ModuleChoices
Total:	Group By	Group By	Count
Sort:			
Show:	☑	☑	☑
Criteria:			
or:			

Total Row

When in Datasheet view, Access provides a quick way to display totals, averages etc at the bottom of the fields. If you need to do this frequently, consider developing a report for this purpose instead (see later).

1 In datasheet view select **Home > Records > Totals** to enable the extra row.

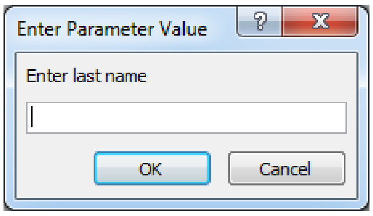
2 Select the appropriate function from the drop-down menu at the bottom of the relevant field. The options will depend on the data type.



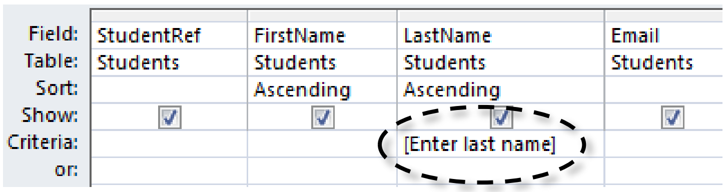
Student #	Module Name	Department Nam	Mark
257846	Quantum Theory for Beginners	Chemistry	54
268549	Quantum Theory for Beginners	Chemistry	
257846	Thermodynamics Theory	Chemistry	
257846	The Secret Life of Metals	Chemistry	49
268549	The Secret Life of Metals	Chemistry	66
268547	Relational Database Design	Computer Science	
278795	Relational Database Design	Computer Science	45
278795	AJAX Development	Computer Science	35
225784	Great Works of English Literature	English	78
225784	Political Comment in Mills and Boon	English	66
268571	Political Comment in Mills and Boon	English	71
Total	11		58

12 ~ Parameter Queries

Filtering criteria are normally saved within a query, but sometimes it is useful to only specify the criteria when the query is run. For example, a query that returns information related to a student could be designed to prompt for a last name when it is run, then only return results that match the entered name.



It is done by entering text enclosed in square brackets into the criteria row of the query. When the query is run, it is this text that appears as the onscreen prompt.



Field:	StudentRef	FirstName	LastName	Email
Table:	Students	Students	Students	Students
Sort:		Ascending	Ascending	
Show:	✓	✓	✓	✓
Criteria:			[Enter last name]	
or:				

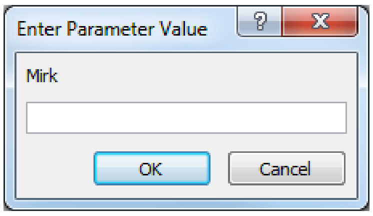
Note: This technique becomes more useful when using Forms. The query can take its input criteria from a form field instead of from this simple dialogue box.

Unexpected parameter requests

If a request for the entry of a parameter value appears when you don't expect it – you know you have not configured a parameter query – this generally means Access does not recognise one or more of the field names used in the query.

With more complex queries this becomes more likely, as expressions frequently require field names to be typed in by hand.

Check the text on the parameter request pop-up as this will indicate which field name is not being recognised.



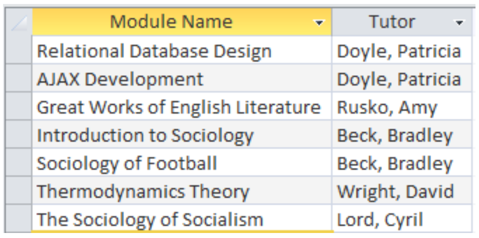
A parameter request has appeared because **Mark** has been misspelled as **Mirk**

13 ~ Alternative Joins

When a query contains two tables with related records, the default relationship (an **inner join**) will only display corresponding records from both tables. Records that exist in one table but have no corresponding records in the second table will not be displayed.

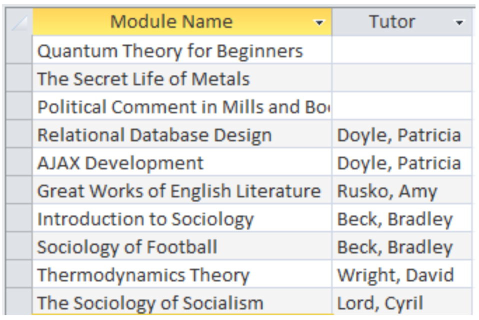
For example, this query shows a list of modules and the tutors. When configured as an inner join, only modules for which tutors have been allocated are shown; when an **outer join** is used, all modules are listed, including those with no tutor.

Inner join



Module Name	Tutor
Relational Database Design	Doyle, Patricia
AJAX Development	Doyle, Patricia
Great Works of English Literature	Rusko, Amy
Introduction to Sociology	Beck, Bradley
Sociology of Football	Beck, Bradley
Thermodynamics Theory	Wright, David
The Sociology of Socialism	Lord, Cyril

Outer join



Module Name	Tutor
Quantum Theory for Beginners	
The Secret Life of Metals	
Political Comment in Mills and Bo	
Relational Database Design	Doyle, Patricia
AJAX Development	Doyle, Patricia
Great Works of English Literature	Rusko, Amy
Introduction to Sociology	Beck, Bradley
Sociology of Football	Beck, Bradley
Thermodynamics Theory	Wright, David
The Sociology of Socialism	Lord, Cyril

13.1 - Configuring an outer join in a query

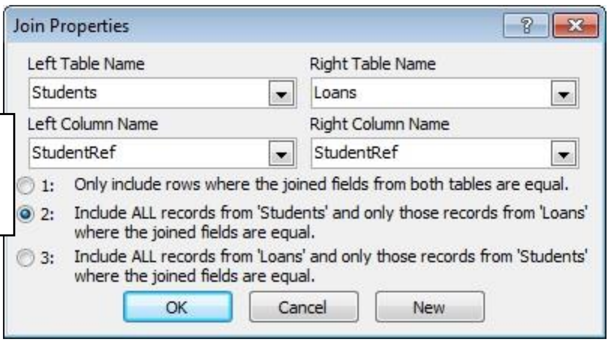
An outer join may be configured when the join is first created or by later editing. In most cases an outer join will only be required in specific queries, in which case it is best configured in the query design.

If an inner join already exists, the reconfigured outer join will apply only to that query and will not affect an underlying inner join.

1 In the query Design view, right-click on the relationship to be changed and choose **Edit Relationship...**, or double-click on the relationship.

2 Modify the relationship settings as required in the **Join Properties** dialogue.

Choose the option that displays the appropriate combination of records, then select OK



Join Properties

Left Table Name: Students
Right Table Name: Loans
Left Column Name: StudentRef
Right Column Name: StudentRef

1: Only include rows where the joined fields from both tables are equal.
[✓ 2: Include ALL records from 'Students' and only those records from 'Loans' where the joined fields are equal.]
3: Include ALL records from 'Loans' and only those records from 'Students' where the joined fields are equal.

14 ~ Action Queries

Queries that retrieve, filter and sort data from underlying tables, but do not change the data, are called **select** queries. **Action** queries are able to modify data and delete records.

Note: Changes made by an Action query cannot be undone so you should take great care when using them.

There are four sorts of action query available:

Make Table	Creates a new table containing the data selected by the query
Append	Adds data selected by the query to an existing table – the field properties and field order must match exactly
Update	Changes the value in one or more fields for selected records
Delete	Removes entire selected records (not just selected data) from the table Deleted records cannot be recovered – there is no undo

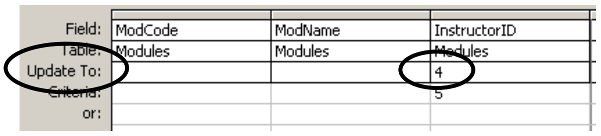
14.1 - Constructing an action query

As action queries can result in significant changes to data, this approach is recommended:

1 First construct a select query that selects the records that need to be affected by the action query. ***Check this is correct before proceeding.***

2 Choose the required action query from **Design > Query Type > ...**

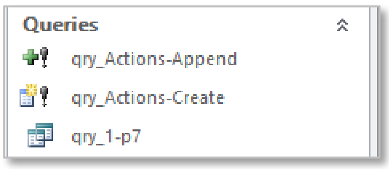
3 Supply any additional information as necessary – this will depend on the type of action query chosen (see below)

Make Table	A dialogue requests a name for the new table
Append	A dialogue requests the name of the existing table to which the data is to be added
Update	Enter value to update to in additional Update To: row  Field: ModCode | ModName | InstructorID Table: Modules | Modules | Modules Update To: | | 4 Criteria: | | 5 or:
Delete	No further editing needed – selected records will be deleted

14.2 - Using action queries

If an action query is to be used on more than one occasion, it is worth saving the design. When saved, an alternative icon is used to indicate an action query.

append query
create query
select query



Opening to edit the design

To open an action query to modify the design, locate the query in the Navigation pane, **right-click** on the query and select Design view.

NEVER double-click on an action query to open for editing.

If the action query is open in **Design** view, switching to **Datasheet** view will usually be possible, and will not result in data changes.

Running an action query

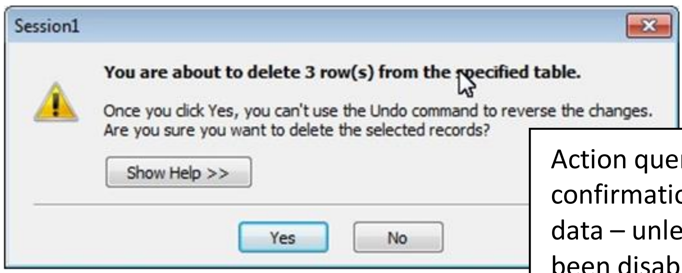
To run an action query either:

- In Design view, choose **Design > Results > Run**

  Or

- From the Navigation Pane, **Right-click** on the query and select **Open.**

  Unless the database designer has disabled messages, you will be warned of any changes to data about to take place and have the option of cancelling the action.



Session1

You are about to delete 3 row(s) from the specified table.

Once you click Yes, you can't use the Undo command to reverse the changes.
Are you sure you want to delete the selected records?

Action queries usually require confirmation before changing data – unless messages have been disabled by the designer.

Append and Delete

One use for action queries is archiving old data, which is often preferable to deleting.

There is no single 'archive query'; instead you must use an append query to identify and add records to a separate archive table (which you can initially create using a make table query), and follow this with a delete query (with the same criteria) to remove them from the original table.