Parameter Queries

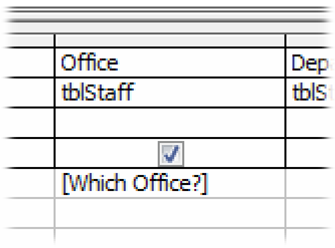
A useful feature of the query is that it can be saved and used again and again, whenever we want to ask the same question. The result we see (the *dynaset*) always reflects the most up-to-date information in the database because what you save is the question, not the answer. You just ask the question again by running the query.

Sometimes we want to ask a question time and time again, but the details (the query's *criteria*) may vary. It would be handy to have a way to run a query and make changes to its criteria without having to design a completely new one from scratch. Access has a tool to solve that problem, the *Parameter Query*.

In fact, the parameter query can be any sort of query. You just employ the methods described here to design the criteria. When you run a parameter query Access presents you with a dialog box prompting you for the parameter value, which it enters into the appropriate criteria cell. You can have as many parameters as you like in a single query. Here's how it's done…

Entering a Parameter

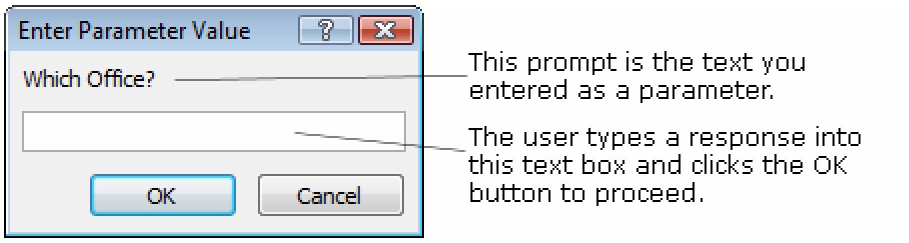
Instead of typing a value or expression into the criteria cell, type some text enclosed in square brackets (**[ ]**) (*Fig. 1*).



Office	Dep
tblStaff	tblS
✓	
[Which Office?]	

Fig. 1 Entering the parameter criteria.

The text you type will appear as a prompt on a dialog box, so you might want it to be in the form of a question to the user. In this example the user will be prompted to type the name of an office when they run the query (*Fig. 2*). The text that the user types will be used as the criteria for that particular field. If the user were to type *London* then this query would display all the records with the entry *London* in the *Office* field. The dialog box looks like this…



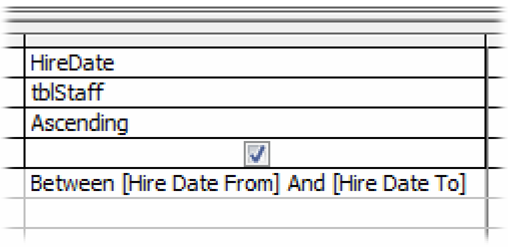
This prompt is the text you entered as a parameter.

The user types a response into this text box and clicks the OK button to proceed.

Fig. 2 Access displays a parameter dialog when the query is run.

Using Multiple Parameters

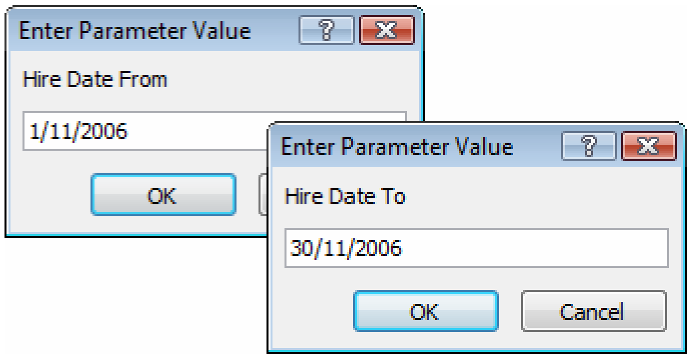
You can enter a parameter almost anywhere you would place a piece of text, number or date in a regular criterion. For example, supposing you wanted the query to prompt the user for two dates to define a date range...



HireDate
tblStaff
Ascending
✓
Between [Hire Date From] And [Hire Date To]

Fig. 3 Using multiple parameters

Instead of typing the actual beginning and end dates into the criteria cell, type a prompt in square brackets. After entering the first date the user will see a second dialog box (*Fig. 4*). After entering dates the query proceeds, inserting the dates into the appropriate places in the criteria expression. In this example the query would display all the record which contained dates in the range *1 November 2006 - 30 November 2006* in the *HireDate* field.



Hire Date From: 1/11/2006

Hire Date To: 30/11/2006

Fig. 4 Access displays a second dialog box.

You can use as many parameters as you want, in as many fields as necessary. The dialog boxes appear in the same order as they do on the QBE grid.

Combining Parameters with Wildcards

A useful feature of the query is its ability to accept wildcards (i.e. an asterisk "*" representing any string of characters; one or more question marks "?", each representing a single character). Wildcards allow you a degree of flexibility when specifying criteria. When you don't know exactly what you are looking for you can use wildcards to give the query a "clue".

This method can also be applied to parameter queries, but you need to do a bit more than just add an asterisk or question mark. The correct syntax is as follows…

For a single wildcard:

Like [type prompt here] & "*"

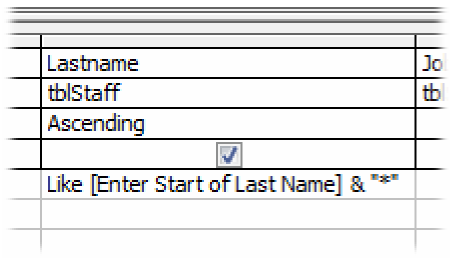
For two wildcards:

Like "*" & [type prompt here] & "*"

When using a single wildcard it can be placed before or after the prompt. You can use asterisks or question marks, or a combination of both.

Using a Single Wildcard

In this example (*Fig. 5*) a single wildcard has been used, an asterisk.



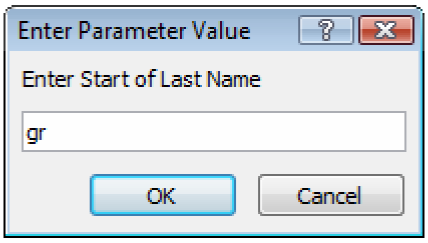
Lastname	Jo
tblStaff	tbl
Ascending	
✓	
Like [Enter Start of Last Name] & "*"	

Fig. 5 Using a wildcard with a parameter.

The parameter…

Like [Enter Start of Last Name] & "*"

…creates a prompt in which the user can enter the first letter or string of letters of the names they want to see (*Fig. 6*).



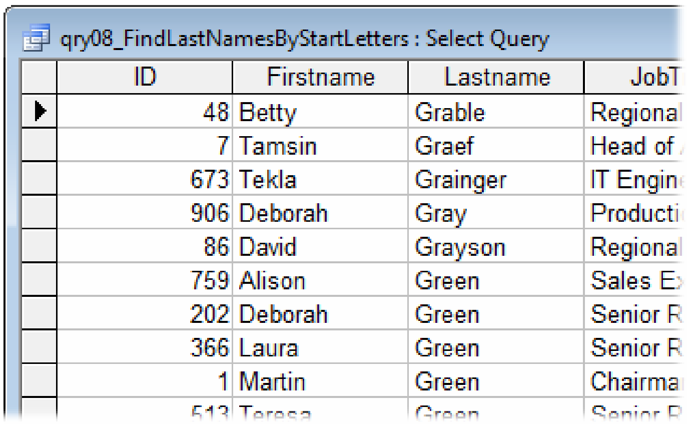
Enter Parameter Value

Enter Start of Last Name

gr

Fig. 6 The user can enter the start letters of a name.

The user has entered the text "*gr*", causing the query to select records with entries in the *Lastname* field of any length starting with the letters "gr". Here's the result (*Fig. 7*) ...



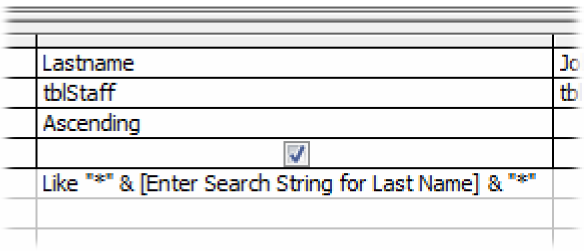
qry08_FindLastNamesByStartLetters : Select Query

ID	Firstname	Lastname	JobT
48	Betty	Grable	Regional
7	Tamsin	Graef	Head of
673	Tekla	Grainger	IT Engin
906	Deborah	Gray	Producti
86	David	Grayson	Regional
759	Alison	Green	Sales E
202	Deborah	Green	Senior R
366	Laura	Green	Senior R
1	Martin	Green	Chairma
513	Teresa	Green	Senior R

Fig. 7 The query returns names starting with the specified letters.

Using Two Wildcards

In this example a two wildcards have been used, both asterisks.



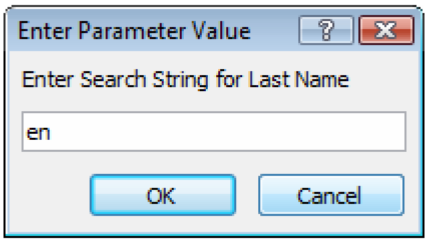
Lastname	Jo
tblStaff	tb
Ascending	
✓	
Like "*" & [Enter Search String for Last Name] & "*"	

Fig. 8 This parameter includes two wildcards.

The parameter…

Like "*" & [Enter Search String for Last Name] & "*"

…creates a prompt in which the user can enter a letter or string of letter that should occur anywhere in the names they want to see (*Fig. 9*).



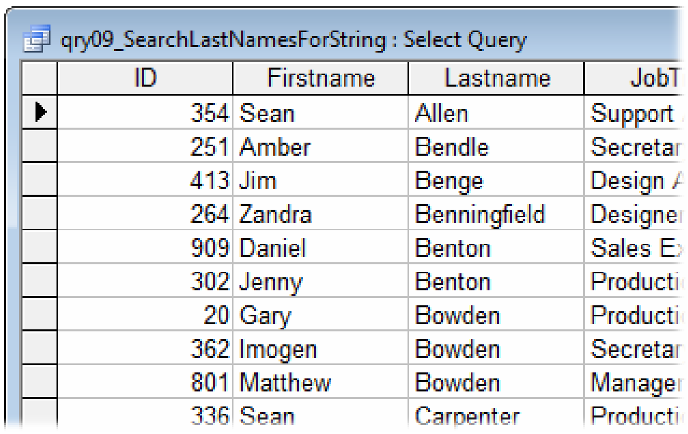
Enter Parameter Value

Enter Search String for Last Name

en

Fig. 9 The user enters a string which must occur in the name.

The user has entered the text "*en*", causing the query to select records with entries in the *Lastname* field of any length containing with the letters "en" together. Here's the result (*Fig. 10*) ...



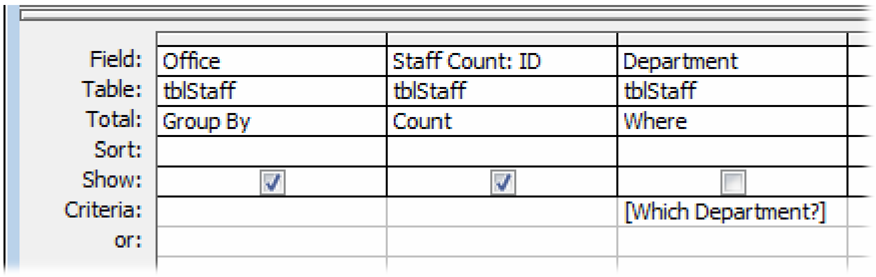
qry09_SearchLastNamesForString : Select Query

ID	Firstname	Lastname	JobT
354	Sean	Allen	Support
251	Amber	Bendle	Secretar
413	Jim	Benge	Design
264	Zandra	Benningfield	Designe
909	Daniel	Benton	Sales E
302	Jenny	Benton	Producti
20	Gary	Bowden	Producti
362	Imogen	Bowden	Secretar
801	Matthew	Bowden	Manage
336	Sean	Carpenter	Producti

Fig. 10 All the results contain the specified string.

Parameters in Totals Queries

When you use the Query Totals tool to summarise your data you specify criteria by adding an extra column in the query design grid and set the value of the **Total** row to *Where*. Then you can specify your criteria as normal in that column. You can use a parameter here too if you wish (*Fig. 11*).

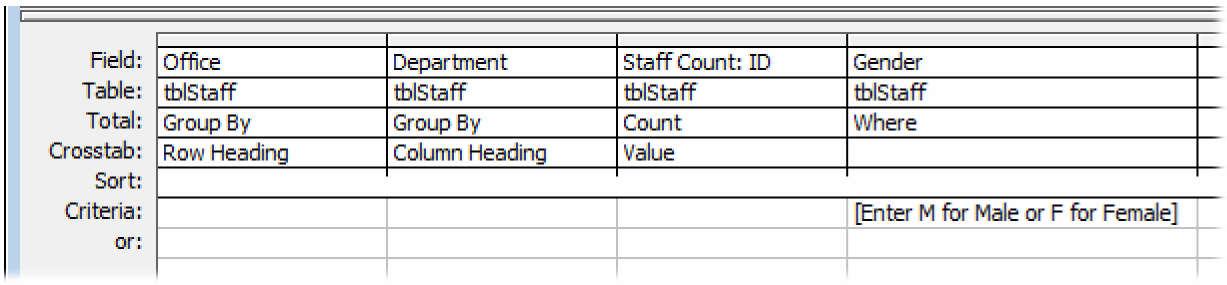


Field:	Office	Staff Count: ID	Department
Table:	tblStaff	tblStaff	tblStaff
Total:	Group By	Count	Where
Sort:			
Show:	✓	✓	☐
Criteria:			[Which Department?]
or:			

Fig. 11 Using a parameter in a Totals query.

Parameters in Crosstab Queries

As with Totals queries, Crosstab queries require you to add an additional column for criteria with its **Total** row set to *Where*. Add the parameter to this column in the usual way (Fig. 12). However, in a Crosstab query you *must* also define the parameters in the **Query Parameters Window** (see: *Using the Query Parameters Window…* below)



Field:	Office	Department	Staff Count: ID	Gender
Table:	tblStaff	tblStaff	tblStaff	tblStaff
Total:	Group By	Group By	Count	Where
Crosstab:	Row Heading	Column Heading	Value	
Sort:				
Criteria:				[Enter M for Male or F for Female]
or:				

Fig. 12 Using a parameter in a Crosstab query

Asking the Questions in the Right Order

When you create a query using more than one parameter, the user sees the prompts in the order that the fields are arranged in the design view of the query, reading from left to right. You normally arrange the fields in the way in which you want to see the results displayed. But what if you want the prompts to appear in a different order? Get to know the **Query Parameters Window**…

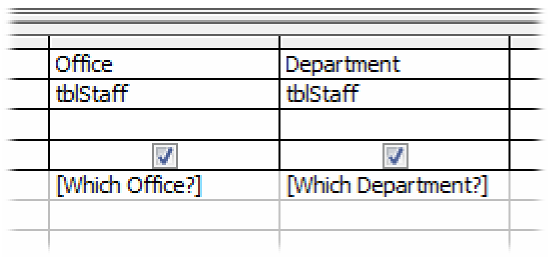
Using the Query Parameters Window…

To control the order in which the prompts appear when running a parameter query containing more than one parameter, you can specify the desired order in the **Query Parameters** window. Here's how...

1. In the query design view choose **Query > Parameters...** to open the **Query Parameters** window.
2. In the **Parameter** column, type the prompt for each parameter *exactly as it was typed in the QBE grid*.
3. In the **Data Type** column specify the kind of data (as defined in the table properties). Pick a type from the list. The default type is *Text*.
4. List the parameters in the order in which you want the dialog boxes to appear when the user runs the query.

Click **OK** to accept your entries and close the window.

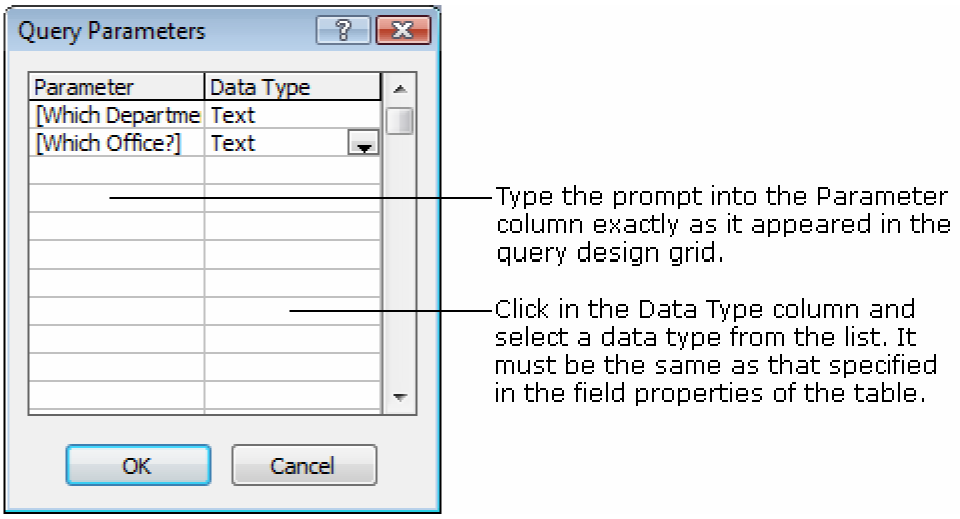
Here is an example of a query containing two parameters...



Office	Department
tblStaff	tblStaff
✓	✓
[Which Office?]	[Which Department?]

Fig. 13 Prompts normally appear in the same order as on the grid.

If you didn't specify otherwise, the prompts would appear in the order that the parameters are arranged in the QBE grid reading from left to right. The user would be asked for an *Office* first and then a *Department*. If, however, you make use of the *Query Parameters* window you can choose the order of the prompts. In this example (*Fig. 14*) the parameters have been arranged in a different order so that the user is asked for a *Department* first and then an *Office*.



Parameter	Data Type
[Which Departme	Text
[Which Office?]	Text

Type the prompt into the Parameter column exactly as it appeared in the query design grid.

Click in the Data Type column and select a data type from the list. It must be the same as that specified in the field properties of the table.

Fig. 14 Use the Query Parameters dialog to specify the order of the prompts.

There in no need to make entries in the *Query Parameters* window if you are happy with the way the query runs, *unless* you are creating a *Crosstab Query* containing parameters, in which case you must enter the details of the parameters to make the query run correctly.

Dealing with Null Entries

If the user fails to make an entry in the parameter dialog box you might expect the query to show all the available data for that field but the converse is true. By leaving the dialog box empty the user is actually specifying "Null" (i.e. "nothing") so the query will return only those records that have a Null value in that field.

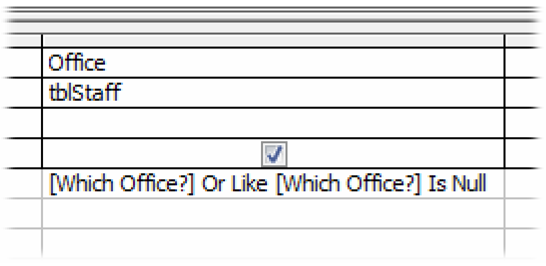
To create a parameter query which returns all records if no parameter value is supplied you need to use a slightly different syntax. Using the *LastName* and *Town* fields again, suppose your rule is that the user must specify a *LastName* but entering a *Town* is optional. If they enter a *Town* they will get records for the specified *LastName* in that town but, if they don't enter a town, they will get all available records for the specified *LastName*.

The syntax for the optional field is as follows:

[Which Office?] Or Like [Which Office?] Is Null

This looks like nonsense but the Access query tool understands what you mean. You can use the syntax simultaneously on more than one field. In fact, if you return to the query design view after running the query for the first time you might see that Access has re-interpreted your instructions and added new criteria to the grid (you could have done this yourself but this way is easier).

It is very important that when you enter the parameter criteria the repeated prompt is written exactly the same way both times (*Fig. 15*). You might like to copy and paste to make sure.



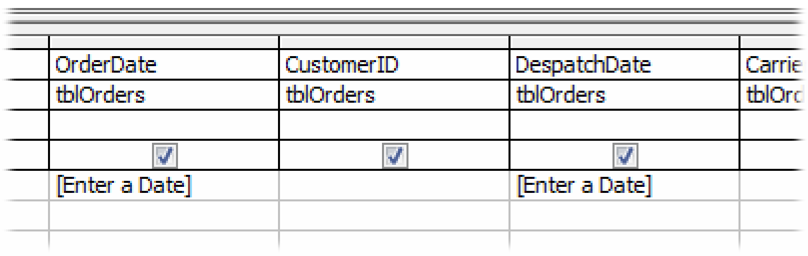
Office
tblStaff
✓
[Which Office?] Or Like [Which Office?] Is Null

Fig. 15 Allowing the user to see all records when the parameter dialog is left empty.

This method works fine as it is but note that when returning all records it will also return those containing null values. If you don't want this to happen add another column to the grid for the same field and set its criteria to **Is Not Null**.

Re-Using the Same Parameter Value

If you enter *exactly* the same prompt text for more than one parameter Access will only ask the question once, and then enter the user's response into each of the criteria. For example (*Fig. 16*) when analysing orders you might want to know which orders were both taken and dispatched on a particular day.



OrderDate	CustomerID	DespatchDate	Carrie
tblOrders	tblOrders	tblOrders	tblOrc
✓	✓	✓	
[Enter a Date]		[Enter a Date]	

Fig. 16 Access will prompt for a date once and use it for both fields.

Dos and Don'ts for Parameter Queries

Prompts Mustn't Match Field Names

When you are designing a parameter query, make sure that the prompt is not exactly the same as one of the field names. You might be tempted to enter the parameter **[LastName]** to prompt the user to enter a name into the dialog box for the *LastName* field. This won't do! Because Access uses field names in square brackets for calculations in queries, your entry will not be recognised as a parameter and will not appear as a prompt. If you really want to use the name of the field as a prompt, a simple solution is to turn it into a question by adding a question mark… **[LastName?]**.

Don't Use Illegal Characters

You can type just about anything for the prompt, but you mustn't use the period (**.**) exclamation mark (**!**) square brackets (**[]**)or the ampersand (**&**), everything else is OK.

Don't Write Too Much!

You are only allowed one line of text in the prompt dialog box, which amounts to about 40 to 50 characters depending on what you say. Anything extra just gets cut off. Check your work!

Get Creative!

You can enter a parameter almost anywhere you would normally enter a specific piece of data in your query criteria. Sometimes the syntax (how you write it out) can be a bit tricky, but persevere until you get the result you need. Here are a few examples...

Finding Records for a Specific Year (or Month) from a Collection of Dates

Supposing you have a field called *InvoiceDate* containing a range of dates covering several years. Use the following criteria...

Year([InvoiceDate])=[Choose a year]

...will create a prompt in which the user can type a year number (e.g. *1998*) to see all the records for that year. I you would rather see records for specific months use...

Month([InvoiceDate])=[Choose a month from 1-12])

Note that the prompt tells the user to enter a number for the month. Access doesn't understand month names.

Finding Records for a Specific Month and Year from a Collection of Dates

If you want to be more specific and call for a particular month *and* year, the criteria...

Month([InvoiceDate])=[Choose a month from 1-12] And Year([InvoiceDate])=[Choose a year]

...will present the user with two dialog boxes, the first asking for a month and the second asking for a year.

Creating a List of Records with Dates in the Last So Many Days

You may want to view all the invoices generated in a recent period, such as the last 30 days. The criteria...

>Date()-[The last how many days?]

...means "*today minus how many days*". The user enters a number (e.g. *30*) to see a list of dates since that many days before today. The *Date()* part creates the current date so this query is always up-to-date.

What About Variable Calculations?

You can even include parameters as part of the definition of a new calculated field. (If you want to learn about calculating in Access check out the tutorial *Calculating in Access Queries*)

For example, you have a list of invoices in which there is a field called *TotalGoods* and you need to calculate the discount (or tax or whatever!), but this changes from time to time. You need to create a new calculated field to work out the new figures. Instead of...

Discount: [TotalGoods]*25/100

...which would always calculate a discount of 25% (*note: unlike Excel, Access doesn't understand the % sign*). You could substitute the fixed figure with a parameter...

Discount: [TotalGoods]*[What discount rate - percent]/100

...which would prompt the user to enter a figure representing the required discount.