Using Microsoft Access

Advanced Queries

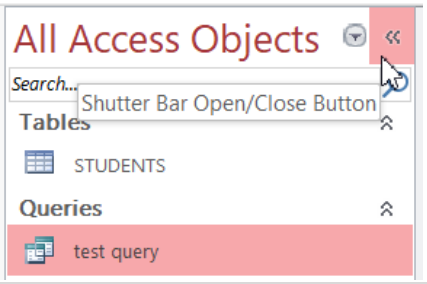
Parameter Queries

In the last section you created queries with various criteria. If you wanted to keep a query with specific criteria you could save it. For example, if you wanted a query that shows all students who live in Dianella, you could create a query with appropriate criteria and save it with a name such as students in dianella. It would get a little complicated, however, if you wanted a query for every suburb, especially if your database contained a dozen different suburbs. Parameter queries are a solution to this problem.

A parameter query allows you to create a query that prompts the user for the criteria when the query is run. You could set up a query so that when it is run, it asks the user to enter a suburb and uses whatever they enter as the criteria. You could use the same query repeatedly without having to modify the design if you want to change the criteria.

Creating a parameter query is like creating a regular select query. The only difference is that where you would normally enter your query criteria, you instead enter square brackets [ ]. This tells access that the user will be prompted for the criteria in this field. If you want to customise the message that is used to prompt the user you can type your own custom message between the square brackets.

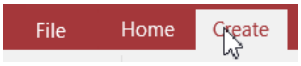
Exercise 1. Creating a Parameter Query



1. Make sure your *Student List* database is open. If you still have your test query open you can close it now. The **Navigation Pane** on the left of the screen will list the objects currently contained in the database which should include one table and one query.

Tip You can hide and display the Navigation pane using the small icon in the top-right corner.

2. Click the **Create** tab on the **Ribbon**.

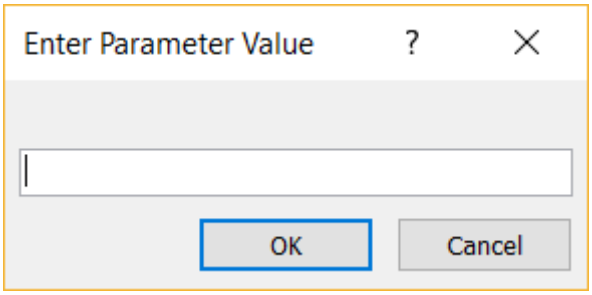


3. Click the **Query Design** icon
4. Double-click, *STUDENTS* in the **Show Table** dialog to add it to the query design and then click **Close** to move to the design window.
5. Adjust your design window as shown in the previous section so that you have plenty of room for selecting fields from your table.
6. Add the following fields to the **QBE Grid**:
    - Last Name
    - First Name
    - Address
    - Suburb
    - Postcode
    - State

7. In the criteria row for the **Suburb** field, add two square brackets [ ] as shown.

Address	Suburb	Postcode
STUDENTS	STUDENTS	STUDENTS
☑	☑	☑
	[]	

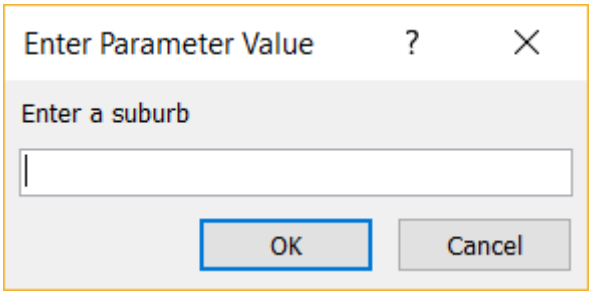
8. Click the **View** icon to view the results of the query. A prompt will appear asking you to enter the criteria for the field.



9. Enter *Dianella* and click **OK**. The query results will show students from Dianella.
10. Click the **View** icon to return to **Design View**.

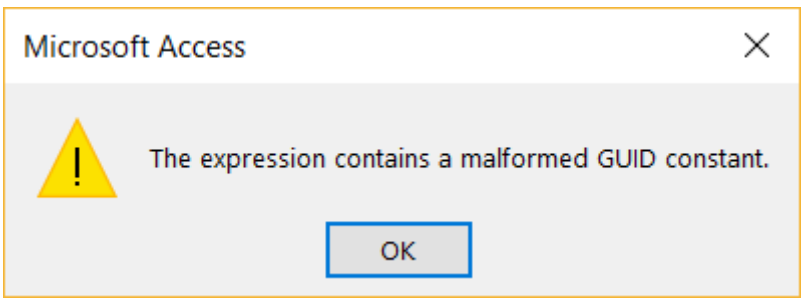
Now we will put a custom message in the criteria prompt. Enter a custom message between the square brackets so that it appears as the following. *[Enter a suburb]*

11. Click the **View** icon again. This time the prompt will include your custom message.



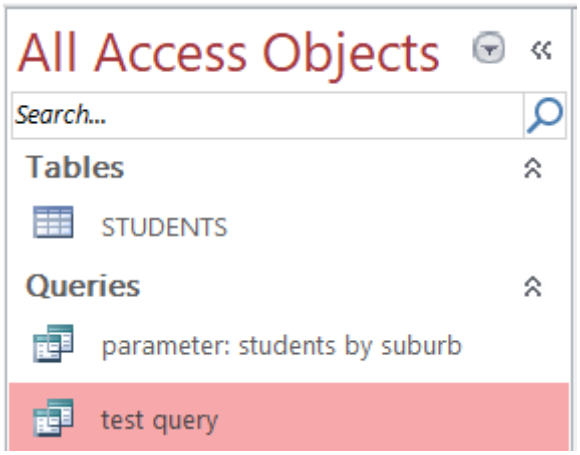
12. Enter *Yokine* and click **OK** (or press **[Enter]**. This time you will see all students from Yokine.
13. Return to **Design View**. Each time you run the query, you could type a different suburb to get different results. All from the same query.
14. Click the Save icon. Name the query *parameter: students by suburb*.
15. Close the query when done.

Note If you try creating a Parameter Query and see a message like the one below, it is because you have used curly brackets { } instead of square brackets [ ].

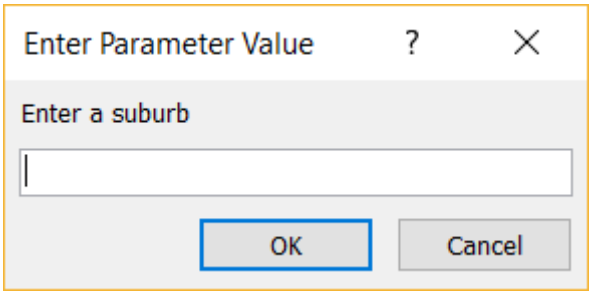


Exercise 2. Using a Saved Parameter Query

Your parameter query should now appear in the Navigation Pane.



1. **Double-click** the *parameter: students by suburb* query to run it.
2. Enter *Bedford* as the suburb and click **OK**. You will see all of the students who live in Bedford.



3. Close the query.
4. **Double-click** the *parameter: students by suburb* query to run it again.
5. Enter *Morley* as the suburb and click **OK**. You will see all of the students who live in Morley.
6. Close the query.

You can run the query with any suburb being used as the criteria without changing the design of the query.

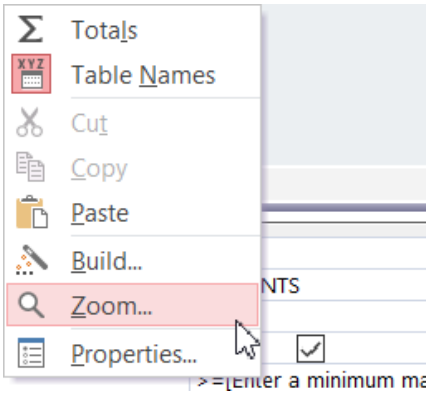
Exercise 3. Creating a Parameter Query with Multiple Criteria

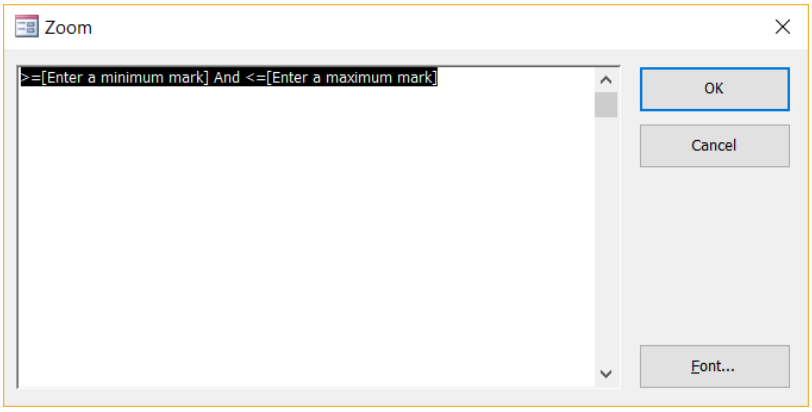
In regular select queries, you can have criteria on as many fields as you like. This applies for Parameter queries as well. If you have more than one parameter criteria in a query then a prompt will appear for each criterion, one after the other before the query results appear. You can also use parameters for range criteria as the following exercise demonstrates. We will create a query that allows the user to enter two amounts, and see all students with marks between those amounts.

1. Create a new query that uses the following fields:
   - Last Name
   - First Name
   - Gender
   - Mark
   - Comment
2. 2) For the *Mark* field, enter the following for the criteria

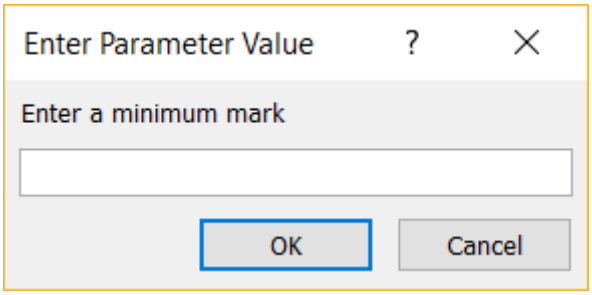
>=[Enter a minimum mark] and <=[Enter a maximum mark]

Tip When you are working with long criteria like this, it can be difficult to edit because of the limited space. To make it easier, **right-click** on the criteria and click **Zoom**. You will then have a nice large window to edit your criteria. When you are done you can click **OK** to confirm the changes.





3. Click the **View** icon to view the results of the query. A prompt will appear asking you to enter a minimum mark.



4. Enter *50* and click **OK**. A second prompt will appear asking you to enter a maximum mark.
5. Enter *70* and clock **OK**. The results will show all students with marks from 50 to 70.
6. Click the **View** icon to return to **Design View**.
7. Save the query as *parameter: student marks between two amounts*.
8. Close the query when it is saved.

Exercise 4. Using Wildcards with Parameters

1. Open the parameter: students by suburb query.
2. When the parameter prompt appears, enter Dian* for the criteria.

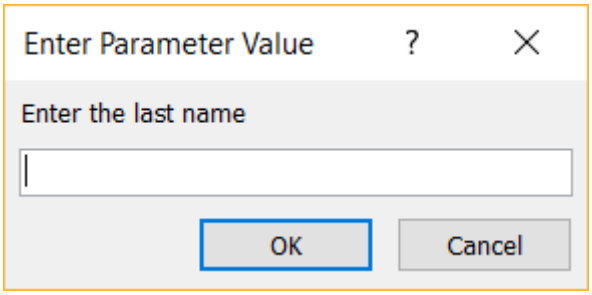
If this criterion were used in a query design, it would result in all students where the suburb name begins with dian. This won't work with parameters though and you will get no results at all. For parameters, the wildcard needs to be built in to the query design along with the parameter.

3. Close the query.
4. Create a new query using the following fields.
    - Last Name
    - First Name
    - Date of Birth
    - Phone
    - Gender
    - Mark
5. For the **Last Name** field, enter the following for the criteria.

*Like [Enter the last name] & **

The **&** symbol will join the user's typed criteria on to a * wildcard. This will mean that the user will only need to enter the first few letters of a last name to get the results.

6. Click the **View** icon to view the results of the query.



7. When the prompt appears enter *S* for the criteria and click **OK**. You will see all students with a last name beginning with the letter S. This makes the query easier for the user.
8. Click the **View** icon to return to **Design View**.
9. Save the query as *parameter: students by last name*.
10. Close the query when it is saved.

Some variations on the above method are shown below.

Field	Criteria	Result
Last Name	Like * & [Last Name] & *	Adds a wildcard before and after what the user types
Phone	Like * & [Phone number]	Phone numbers ending with the numbers the user types

11. Create queries to test each of the two examples shown.

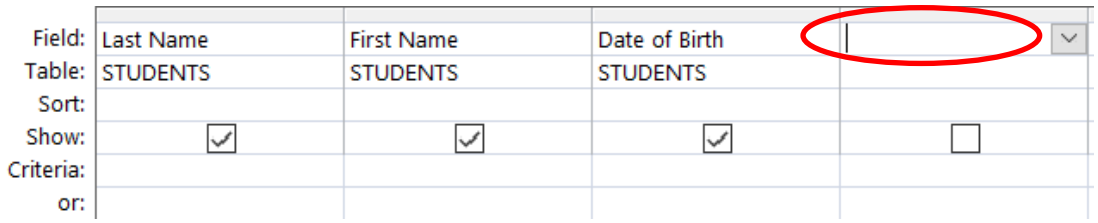
Calculated Fields

Queries can include additional fields which automatically calculate a result based on existing fields. The calculations in these fields are similar to the calculations in **Excel** formulae. In the following exercise we will create a basic calculated field to work out a student's age from their Date of Birth. The following guidelines apply to calculated fields.

- Like Excel, calculations in access follow the order of operations rules. I.e.
  - Anything contained within brackets is calculated first
  - Indices (^) are calculated next E.g. 4^2 means 4 to the power of 2
  - Multiplication and division calculated next
  - Addition and subtraction calculated last.
- When you are referring to a field in your calculation, the field name must be enclosed in square brackets. E.g. [Mark].
- You can't combine different data types in a calculation. E.g. You can't add a number field to a text field.

Exercise 5. Creating a Calculated Field

1. Create a new query with the following fields:
   - Last Name
   - First Name
   - Date of Birth
2. Click in the blank field after *Date of Birth* as shown.



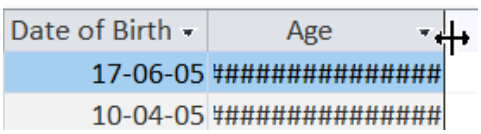
The calculation is typed where the field name would normally go. In an Excel calculation, you would begin with an equal sign. In an Access calculated field, you begin with a name you want the calculated field to have, followed by a colon. This calculated field is to calculate a student's age.

3. Enter the following in the **Field** row (you may like to use the **Zoom** option shown previously when you are editing the calculation).

Age: (Date()-[Date of Birth])/365

4. This calculation will subtract their date of birth from the current date *date()* to calculate their age in days. The result is then divided by 365 to get the age in years.
5. Click the **View** icon to view the results of the query.

Note If your **Age** field shows the results as a line of # symbols, that isn't an error. It simply means that the column isn't wide enough to display the whole number. You can re-size the column to make it wide enough to fit the numbers.



The calculation will appear as an additional field, with the result appearing for each student.

6. Click the **View** icon to return to design view.
7. Save the query as *student listing with age*.

Exercise 6. Formatting a Calculated Field

In the previous exercise, the resulting calculated field contained ages with a large number of decimal places. You can customise fields in a query by editing properties that are very similar to the ones found in table design.

Query1

Last Name	First Name	Date of Birth	Age
Robbins	Mark	17-06-05	13.6493150684932
Stevens	Sarah	10-04-05	13.8356164383562
Andrews	Claire	01-11-05	13.2739726027397

1. Make sure you still have your *student listing with age* query open in **Design View**.
2. **Right-click** on the **Age** field and choose **Properties** (you can also get to properties by pressing **Alt Enter** or by clicking Property Sheet on the **Ribbon** under the **Query Tools, Design** tab).

The **Properties Sheet** will show properties for the age calculated field (you can also edit properties for any regular field in the same way).

3. Click in the **Format** property and change the format to *Fixed*.
4. Click in the **Decimal Places** property and enter *1*.

Property Sheet

Selection type: Field Properties

General | Lookup

Description	
Format	Fixed
Decimal Places	1
Input Mask	
Caption	

5. View the results of the query again. The ages will now be formatted with one decimal place.
6. Save and close the query.

Query1

Last Name	First Name	Date of Birth	Age
Robbins	Mark	17-06-05	13.6
Stevens	Sarah	10-04-05	13.8
Andrews	Claire	01-11-05	13.3
McKay	Tim	02-08-05	13.5

You can close the **Properties Sheet** or leave it open if it's not taking up too much room.

Note The **Properties Sheet** can be positioned on an edge of the window or it can be positioned floating anywhere on the screen. Move it by dragging its title bar

Note In the latest versions of Access, you can create a calculated field as part of your table design

Exercise 7. Using a Function in a Query

Functions in **Access** are similar to functions in **Excel** and can be used to simplify complex calculations. The following example demonstrates the use of the *Immediate If* function which is similar to Excel's *If* function.

1. Create a new query with the following fields:
   - Last Name
   - First Name
   - Mark
2. In the next blank field, enter the following.

Pass: IIf([Mark]>=50,"Pass","Fail")

The first part will be the name of the new field (Pass).

This is followed by the IIF. Like all functions, its components are enclosed in brackets. This type of function has three parts separated by commas.

The first part specifies the criteria. In this case *[Mark]>=50* which asks if the contents of the Mark field are greater than or equal to 50.

The second part specifies what the answer will be when the condition is true (display the text *"Pass"*). When text is being referred to in a calculation it always needs to be enclosed in " ".

The third part specifies what the answer will be when the condition is false (display the text *"fail"*).

3. Click the **View** icon to view the results of the query.

Query1

Last Name	First Name	Mark	Pass
Robbins	Mark	78	Pass
Stevens	Sarah	62	Pass
Andrews	Claire	58	Pass
McKay	Tim	34	Fail

Any student with a mark less than 50 will have *Fail* in the *Pass* field.

4. Click on the mark for *Laura Davies*.
5. Change the mark to *48*. As soon as you move on to a different record, the *Pass* column will update.
6. Make sure you are still in the *Mark* column and click the Descending icon so that the highest marks are at the top. Changes to the sort order will be saved as part of the query design.
7. Save the query as *student marks* and close the query when done.

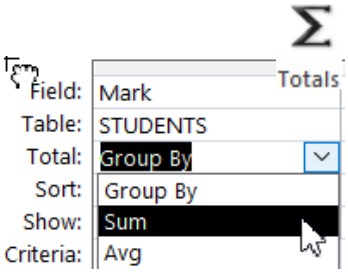
Note when you are viewing the results of a query you are actually viewing a selection from the table. When you change data in a query's datasheet view those same changes are being made to the table data. Any formula depending on that data will automatically update.

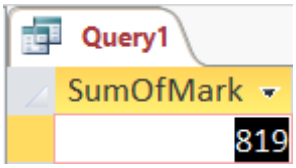
Summary Queries

Unlike other queries you have done, a summary query won't show individual records. It will only show a summary in the form of averages, totals and other available calculations. **Summary Queries** are useful for reporting where you are only interested in a summary of the information without the detail.

Exercise 8. Creating a Query with a Total

1. Create a new query with only the **Mark** field.
2. To change to a **Summary Query**, click the **Totals** icon on the **Ribbon**.
3. Your **QBE Grid** will now include a **Total:** row.
4. Change the **Total** figure for the **Mark** field from **Group By** to **Sum**.
5. View the results of the query. Because of this change the query result will now show the Sum (total) of the Mark field instead of showing each record.

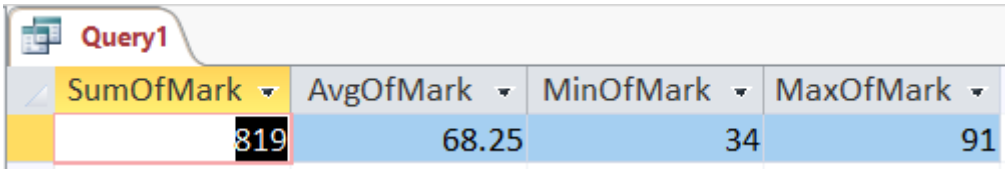




6. Return to **Design View**.
7. Add the **Mark** field to the **QBE Grid** three more times. Change the total row for each one to *Avg*, *Min* and *Max* as shown below.

Field:	Mark	Mark	Mark	Mark
Table:	STUDENTS	STUDENTS	STUDENTS	STUDENTS
Total:	Sum	Avg	Min	Max

8. View the results of the query. The results will now show the **Sum** of the Mark field, the **Average** of the mark field, the **Minimum** amount in the Mark field and the **Maximum** amount in the Mark field.

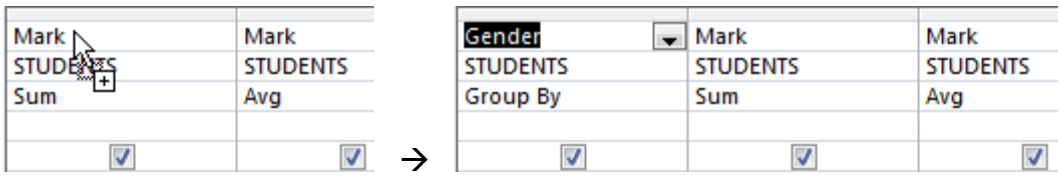


9. Return to **Design View**.
10. Save the query as *summary: student marks*.

Exercise 9. Using Grouping in a Summary Query

When using Summary Queries it is common to group the results by a particular field instead of viewing the totals for the entire query.

1. Drag the **Gender** field on to the first **Mark** field. This will insert the **Gender** field to the left of the first **Mark** field.
2. Leave the **Total** row for the **Gender** field as *Group By*.



3. View the results of the Query. Now you will see totals for the *Female* students and totals for the *Male* students.

summary: student marks

Gender	SumOfMark	AvgOfMark	MinOfMark	MaxOfMark
Female	414	69	48	91
Male	405	67.5	34	83

4. Return to **Design View**.
5. Add the **Suburb** field as shown below.

Field:	Gender	Suburb	Mark	Mark	Mark	Mark
Table:	STUDENTS	STUDENTS	STUDENTS	STUDENTS	STUDENTS	STUDENTS
Total:	Group By	Group By	Sum	Avg	Min	Max

6. View the results of the query. The results will now show the totals grouped by Gender and grouped by Suburb.

summary: student marks

Gender	Suburb	SumOfMark	AvgOfMark	MinOfMark	MaxOfMark
Female	Bedford	91	91	91	91
Female	Dianella	155	77.5	72	83
Female	Morley	106	53	48	58
Female	Yokine	62	62	62	62
Male	Dianella	246	61.5	34	78
Male	Morley	76	76	76	76
Male	Yokine	83	83	83	83

7. Return to **Design View** and display the **Properties Sheet**.
8. Click in the first **Mark** column (the one with the **Total** set to *Sum*).
9. Change the **Caption** property to *Total*. This column will now show *Total* where the field name would usually go instead of showing *SumOfMark*.
10. Change the caption property for the other **Mark** columns so that the query results look like the following.

Gender	Suburb	Total	Average	Lowest	Highest
Female	Bedford	91	91	91	91
Female	Dianella	155	77.5	72	83

11. Save and close the query.