

Microsoft Access 2007 – Calculations

We often need to perform calculations with our Access data and usually it is good practice to ask Access to do the calculations for us rather than storing the results of calculations that were performed elsewhere. Calculations are best regarded as information that we derive from our stored data, rather than being stored as data themselves. This is because storing data that could be created from existing data is both wasteful of time and space and potentially inaccurate. For example, there is no need to store a person's **Full Name** if we have fields that contain their **Title**, **First Name** and **Last Name**. Instead, we can bring together the data from the necessary fields to construct the full name whenever we need it. It is usual to store a person's **Birth Date** rather than their **Age** because age changes with time and it is easy to calculate their age when it is required. When recording purchases, we need to store the **Unit Price** and **Quantity** of items on an order but the **Total** is simply created by multiplying these two fields together whenever we need to see it. If we were to store any of this calculated data we would have to remember to re-enter the calculated result if any of the values it was derived from were changed. Failure to do this would lead to errors in our data. When Access calculates values for us we can be sure the results are always up-to-date.

Where and What Can Access Calculate

Queries are most frequently used for organising and summarising data but they can also perform calculations. Calculations can also be created on Forms and Reports. Many mathematical calculations can be performed using simple arithmetic. For more complex work Access, like Microsoft Excel, has a wide range of ready-made functions available. When displayed in forms and queries, calculated values are automatically made read-only. Their values can only be changed by changing the data from which they were derived.

Calculations on Forms

The form illustrated below (*Fig. 1*) is displaying three calculations, all derived from other data displayed on the form (NOTE: the data that you use in your calculation does not have to be visible on the form providing that it is contained in the recordset on which the form is based). In the form's header the person's full name is displayed using the information contained in the FirstName and LastName fields. Further down the form the person's age is shown, calculated from the BirthDate field and the current date. Also shown is the number of Months Service with the company which has been calculated from the HireDate field and the current date.

Field	Value
StaffID	1
FirstName	Martin
LastName	Green
BirthDate	27/09/1950
Age	58
HireDate	01/11/1995
Months Service	161
Gender	M
JobTitle	Chairman
Salary	£103,000
Office	London
Department	Management

Fig. 1 A form displaying calculated data.

To add a calculation to a form, in design view place a new **Text Box** control on the form. The text box will contain the word *Unbound* indicating that it is not linked to any particular field in the data, and will be accompanied by a label whose caption you can change to something more suitable or delete if you don't need it. To insert the calculation select the text box, open its Properties Sheet and enter an equals sign (=) followed by the required expression in the **Control Source** property (on the **Data** tab). Once completed, the expression also appears in the text box itself, where the field name would normally appear. In the second example (*Fig. 2*) a subform, normally displayed as a datasheet on its mainform, displays a total value created by multiplying together the Quantity and Price fields. Here are some more examples of expressions:

- **= [FirstName] & " " & [LastName]**
This expression concatenates (joins together using the **&** character) the information from the FirstName and LastName fields, adding a space in between.
- **=Int((Date()-[BirthDate])/365.25)**
This expression calculates the person's age by subtracting the date in the BirthDate field from today's date (supplied by the **Date()** function) to give their age in days, dividing this by the average number of days in a year (365.25) then rounding the result down to a whole number using the **Int()** function.
- **=DateDiff("m",[HireDate],Date())**
This expression uses the **DateDiff()** function to calculate the difference in months between today's date and date in the HireDate field to show how many months the person has been working for the company.

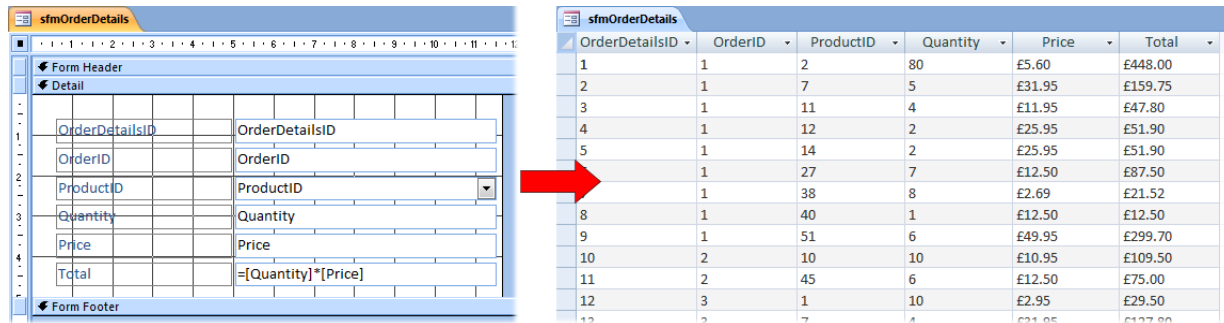


Fig. 2 The expression **= [Quantity]*[Price]** creates a Total value on this subform.

Calculations in Queries

A similar procedure is used to create calculations in queries. When performing a calculation in a query it is usual to create a new column to contain the calculation and display the result. This is often called a *calculated field* because, once created, the new field and its data can be treated like stored data and sorted, summarised or used in further calculations like a normal field. To create the calculated field type an expression in the top row of a new, empty column in the query design grid. You can use the same expressions as you would use on a form but omit the equals sign and, in its place, type a name for the new field followed by a colon (:)(Fig. 3). For example:

- **Total:[Quantity]*[Price]**
Creates a new, calculated field containing the product of the Quantity and Price fields.
- **FullName:[LastName] & ", " & [FirstName]**
Creates a new field containing the data from the LastName and FirstName fields, separated by a comma and space.

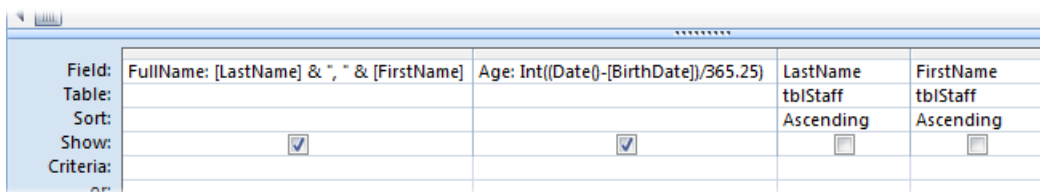


Fig. 3 This query contains new calculated fields in which the Full Name and Age have been created.

You do not have to include the fields used in the calculation as part of the query providing they are present in the query's source data. Running the query displays the result (Fig. 4):

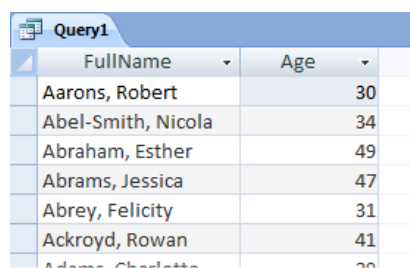


Fig. 4 When run the query displays the calculated values.