Table of contents

Databases

In this chapter, you will learn to:

- Split, compact and convert a database

SPLITTING A DATABASE

All the elements or objects in a database compose in fact only one file (*.MDB). It is possible to export or save tables in another file.

To split a database into two files:

- Select the ***TOOLS*** menu.
- Click the option ***DATABASE UTILITIES.***
- Then choose ***DATABASE SPLITTER*** in the submenu.
- The tables now belong to a second access database.

COMPACTING A DATABASE

As soon as an important quantity of data have been updated or added in a database, the file will probably be fragmented on the disk. The way the data is organized on the disk is far from being optimal.

In order to use less space on the disk, you can compact the database.

- Completely close the database and check that it is not being used by someone else.
- Select the ***TOOLS*** menu.
- Click the option ***DATABASE UTILITIES.***
- Then choose ***COMPACT DATABASE*** in the submenu.
- Access reorganizes your data.

CONVERTING A DATABASE

If you want to use a database from a previous version (For example: from Access 97 to Access 2000 or Access 95 to Access XP), you have to convert the database.

- Select the ***TOOLS*** menu.
- Click the option ***DATABASE UTILITIES.***

- Then choose ***CONVERT DATABASE*** in the submenu.
- You will obtain a converted copy of the database.

In Access 2003, the standard file still is Access 2000.

The Tables

In this chapter, you will learn to:

- Organize tables
- Manage the relationships between tables

EXCHANGING DATA

Tables represent the foundations of a database because they contain the information.

In some cases, these tables have already been created in another application. These data can be imported by using two different methods:

- the import technique (***IMPORT***)
- an attachment (***LINK***)

Importing a table

By using this method, Access will create a copy of the original table and locate it in an existing database.

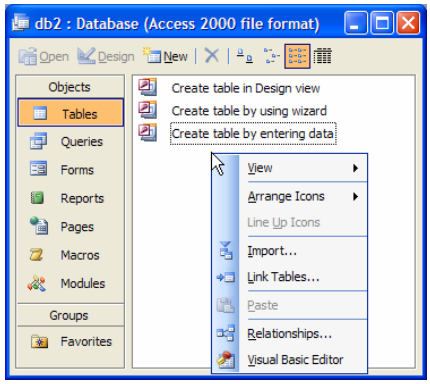
You can choose where the imported data will be stored: in an existing table or in a new one.

In order to use the data in another file:

- Select the ***FILE*** menu.
- Click the option ***GET EXTERNAL DATA.***
- Then choose ***IMPORT*** in the submenu.

Remark :

You can also click with the right mouse button in the ***DATABASE*** window and choose ***IMPORT*** in the contextual menu.

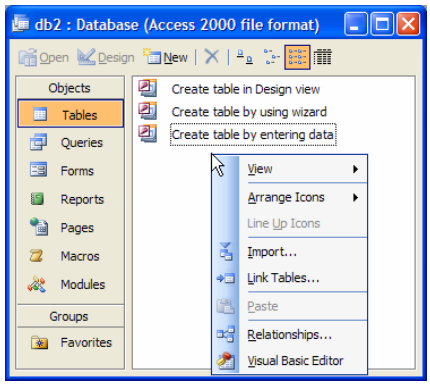


Linking a table

- Select the ***FILE*** menu.
- Click the option ***GET EXTERNAL DATA.***
- Then choose ***LINK TABLES*** in the submenu.

Remark :

You can also click with the right mouse button in the ***DATABASE*** window and choose ***LINK TABLES*** in the contextual menu.



Any modification you carry out in a linked table is updated in the linked table, because you are in fact always working with the original table.

Exporting tables

Select the ***FILE*** menu.

Choose the option ***EXPORT.***

Exporting a table allows the user to copy the Access data in another file. Access makes a copy of the table and stores it in an existing Access database or in another file.

Example: an Access table can be exported in an Excel file in order to use it for a pivot table.

Remark :

Exporting a table does not create a link between the original table and the exported table.

CREATING RELATIONSHIPS BETWEEN TABLES

The creation of relationships between several tables is very important in a relational database.

Relationships between tables can be created:

- Permanently for the whole database
- Temporarily for a query for example

In order to create a relationship between two tables, you must have a common field in both tables. This field does not necessarily have the same name but must contain identical information (same type of data and same size).

If you want to use an AutoNumber field (automatic numbering) as the common field, the second table has to contain a “number” field of the type ***LONG INTEGER***.

Type of relationships between tables

There exist different types of relationships between tables:

ONE-TO-ONE RELATIONSHIP: each row of the first table corresponds to only one row in the second table. Tables with a one-to-one relationship are often merged into one table.

ONE-TO-MANY RELATIONSHIP: each record of the basic table (the one with the primary key) can correspond to several records in the linked table. It is the type of relationship that is used the most.

MANY-TO-MANY RELATIONSHIP: one record of the "A" table can correspond to several records in the "B" table. Moreover, one record of the "B" table can correspond to several records in the "A" table. This type of relationship cannot be achieved directly in Access. You have to use two One-to-Many relationships as well as an intermediate table to simulate a many-to-many relationship.

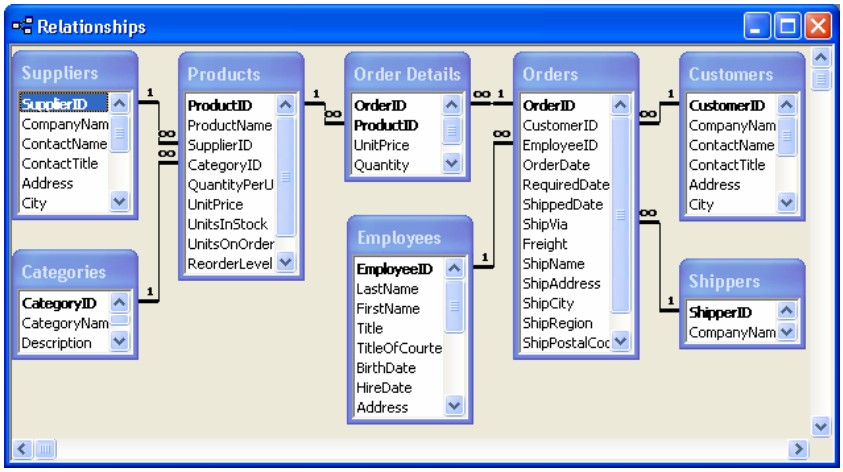
When creating relationships between tables, you can also impose a ***REFERENTIAL INTEGRITY.*** This makes sure your data are coherent. Each time an ‘A’ object refers to a ‘B’ object, the latter has to exist. So that it cannot be deleted. (For example:

Orders of the 'Orders' table refer to a customer in the 'Customer' table. If you delete the latter in the 'Customer' table, some orders will be linked to an unknown customer!)

THE REFERENTIAL INTEGRITY also aims at checking the data entered in a field of the table that refers to a field in another table, so as to avoid referring to a non-existent record. For example, in the "Orders" table if you refer to a non-existent customer, Access will refuse your data as long as the customer in question has not been created.

Viewing the existing relationships

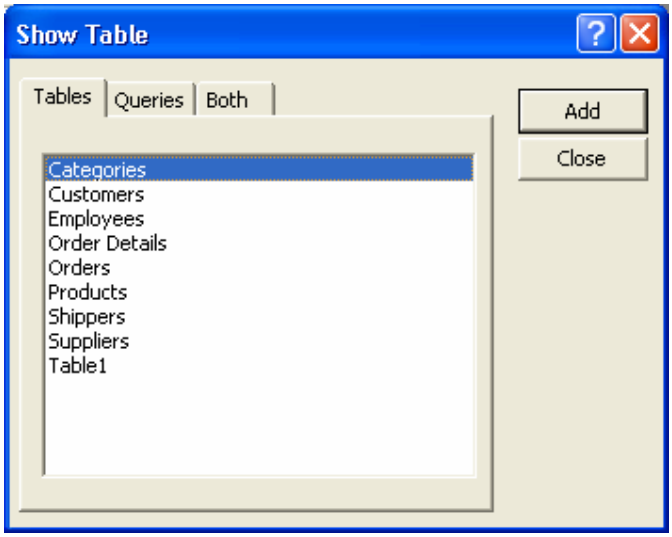
- Select the ***TOOLS*** menu.
- Choose the option ***RELATIONSHIPS*** or click the  icon.
- A screen will be displayed, which is similar to the one below:



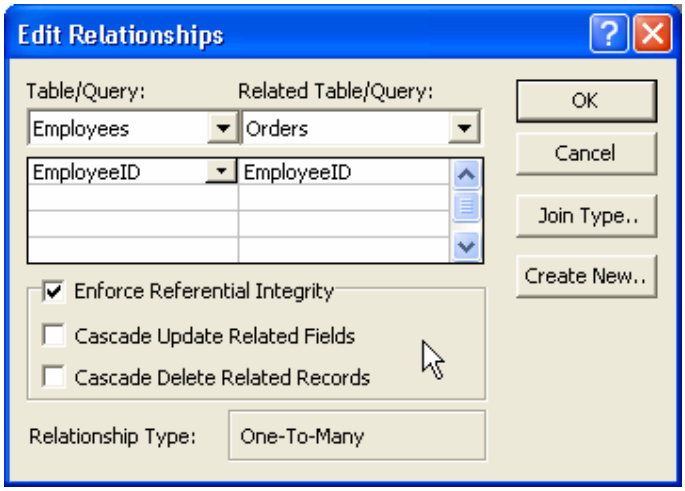
The symbols ***1*** and ∞ define the type of relationship.

Creating relationships

- Select the ***TOOLS*** menu.
- Choose the option ***RELATIONSHIPS*** or click the  icon.
- Click the  button in order to display the tables that contain fields you want to link.



- Select the tables you want to link and click the Add button.
- Close the dialog box.
- Position the cursor (click with the mouse) on a column heading in the table and drag (hold down the left mouse button) the item to a corresponding field in the linked table.
- Check if necessary the options in the ***RELATIONSHIPS*** dialog box and fill them in.



- Click the OK button to confirm the type of relationship.

The Referential Integrity

Using the referential integrity strengthens the relationship and gives access to other options. As soon as it is activated, Access checks that the information contained in the two tables correspond.

If you use a customer number in the order table that doesn't already exist in the customers' table, Access will not accept if you add a new order.

Cascade update and cascade delete

When you activate the referential integrity, two options become available: the cascade update and the cascade delete.

By activating the cascade update function, Access will update the data in the linked table (Child table) when the primary key has been changed in the original table (Parent Table) instead of refusing to carry out the operation.

By activating the cascade delete function, if you delete a record in the original table, Access will also delete all the corresponding records in the linked table. This is to keep the referential integrity of the data. Before the deletion, Access will display a warning message.

> *Example:*
>
> *When you change the customer number in the "Clients" table, automatically the same customer number will be changed in the "Commandes" table.*

Using queries

In this chapter, you will learn to:

- Query your databases

Queries are used in Access for selections, modifications, or statistics…

They query the database but allow the user to make calculations on fields.

In Access, there are two types of queries:

- The Select Queries
- The Action Queries

This chapter deals with advanced select queries (with functions and SQL) as well as the action queries.

SELECT QUERIES

Join types

In the queries based on several tables, the type of join will determine how Access will make the data of several tables correspond. There are two types of joins.

Inner Join

This is the default join type.

When merging two tables (query), Access will only show the records that have a common field in both tables.

When merging data from two tables, Access will only show the records that contain the common field which has the same value in both tables.

> *Example:*
>
> *The information of customers that have never ordered will not be taken into account in the result of a query or report, because the "Commande" table does not contain any reference on these customers.*

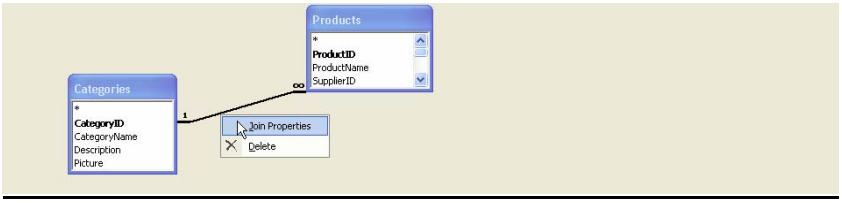
Outer Join (left join or right join)

Access displays all the records of one of the tables, even when there is no corresponding record in the other table.

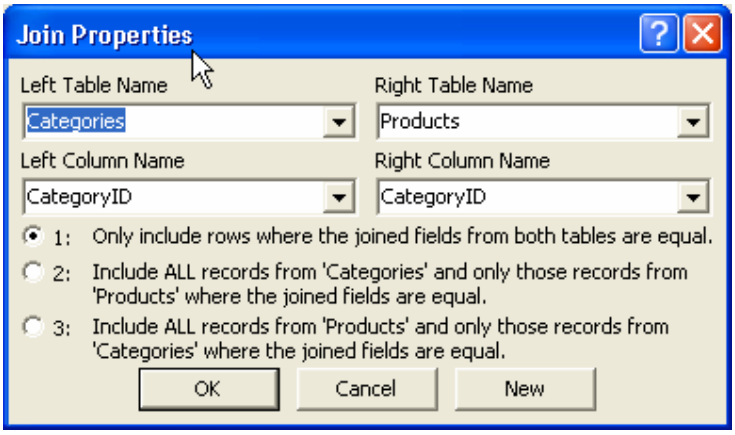
Example:

All the data concerning a customer are displayed in the result, even the information on the customers that have never ordered.

Modifying the join type



If you want to modify the type of join, right-click the link and choose ***JOIN PROPERTIES***.



In our case : if you want to obtain the products per category, but you also want the categories that do not contain any products, select the option 2. The option 3 is not useful in this case, as with the referential integrity there cannot be a product that is not attached to a category.

Calculating with fields

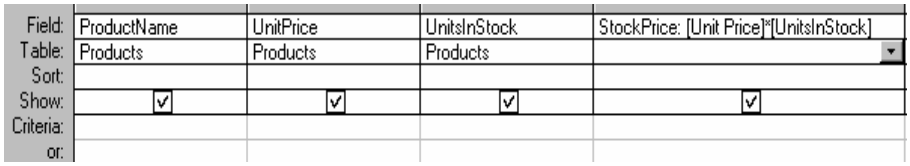
In a query, you can add fields whose value is calculated from the contents of other fields. The calculated fields can contain numbers, dates or even text.

These queries can be combined with parameters.

- Select an empty column
- Type the calculation in a new column in the ***FIELD*** row.

Access will automatically give a name to this field. But you can give a name yourself by entering the name with ":" before the calculation.

Example:



Field:	ProductName	UnitPrice	UnitsInStock	StockPrice: [Unit Price]*[UnitsInStock]
Table:	Products	Products	Products	
Sort:				
Show:	☑	☑	☑	☑
Criteria:				
or:				

Calculating in a query

Type ***[UNIT PRICE]-[UNITSINSTOCK]*** in the field row, and Access will create ***EXPR1: [UNIT PRICE]-[UNITSINSTOCK].***

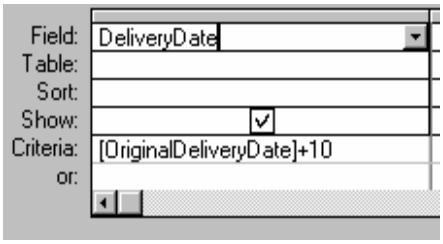
You can then replace **Expr1** by StockPrice (see the example above).

All the classic operators can be used in queries:

Plus	Minus	Multiply	Divide by	Square
+	-	*	/	^

Remark :

Calculations can also be used in the criteria row.



Field:	DeliveryDate
Table:	
Sort:	
Show:	☑
Criteria:	[OriginalDeliveryDate]+10
or:	

The functions

Access proposes a large variety of functions, which can be used for specific types of calculations. You can enter these functions directly or choose them in the "*expression builder".*

Syntax: **functionname(parameters)**

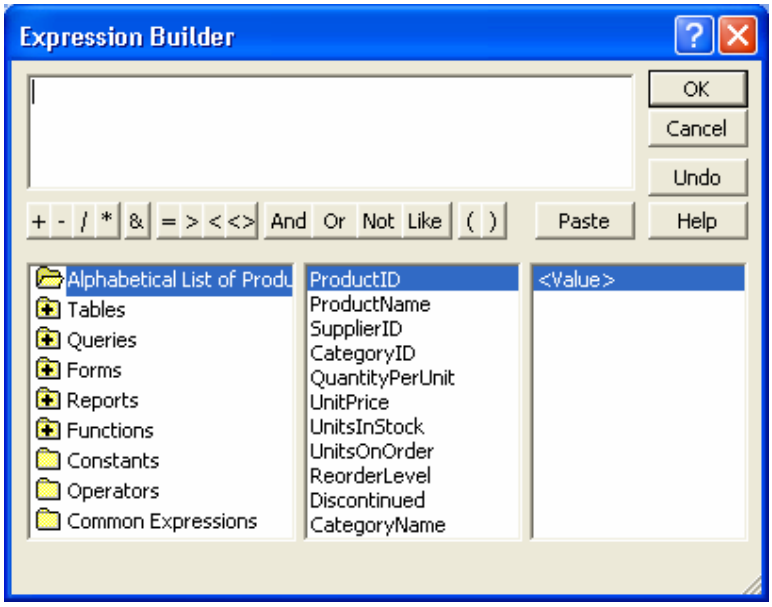
> *Example: Ucase([Lastname])*
>
> *This function changes the 'Lastname' contents into uppercase.*

When you build an expression, the main difficulty is to know the exact syntax of the different functions used in Access. That's why you have an Expression Builder.

To build an expression:

↳ Open the ***EXPRESSION BUILDER*** window by clicking the icon  on the toolbar.

↳ The next screen will be displayed.



The functions are grouped into different categories :

- Date and time
- Text

You can find them via ***FUNCTIONS, BUILD-IN FUNCTIONS***…

Example of function	Description
UCASE([field])	Turns the text into uppercases.
LCASE([field])	Turns the text into lowercase;
LEFT([field]),X)	Takes the first "X" characters (starting from the left) in the field.
RIGHT([field],X)	Takes the first "X" characters (starting from the right) in the field.
MID([field],X,Y)	Takes Y characters in the field starting with the X character
LEN([field])	Gives the length (number of characters) of the field
IIF([field1]>6,[field1],[field2])	The function iif starts with a test (here [field1]>6), followed by the value if the test is true, and the value if the test is false.
DATEDIFF(code,[date2],[date1])	Calculates the duration (in days, weeks, months, …) between two fields with dates. The code determines the interval in question. "d" = days "m"=months "yyyy" = years rem: the function rounds up to the nearest whole number
NOW()	Current date and time (system).
DATE()	Current date

SQL - QUERY

SQL is the abbreviation of STRUCTURED QUERY LANGUAGE.
It is a language that allows the user to manipulate the data of a relational database.

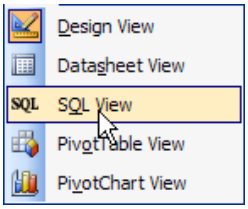
General syntax of a simple select query in SQL

```
Select FieldName1, FieldName2
From TableName
Where Criteria
```

> *Example of a select query with SQL:*
>
> *This example displays as a result the fields "Last Name" and "First Name" of the table "T_Employees". The criteria being that: the salary must be bigger than 100000.*

```
SELECT Lastname, Firstname
FROM T_Employees
WHERE Salary > 100000
```

After having defined a query in the ***DESIGN VIEW***, you can display the SQL code by choosing ***SQL VIEW***.




```
SELECT DISTINCTROW T_Employees.LastName, T_Employees.FirstName
FROM T_Employees
WHERE (((T_Employees.Salary)>100000));
```

In Access, you have the choice; you can create a query by entering the SQL code or by using the Design view.

Union query

Union queries are used for bringing together the results from two different queries.

> *Example:*
>
> *You want a list with all the employees, but at the same time the contact people of your different customers.*

Union queries cannot be created in design view. You have to create them directly in SQL. The syntax is simple, you simply have to separate the two queries by using the UNION keyword.

```
SELECT DISTINCTROW T_Employees.LastName, T_Employees.FirstName,
T_Employees.Address

T_Employees.Postalcode, T_Employees.City
FROM T_Employees

UNION

SELECT DISTINCTROW T_Customers.ContactLastName,
T_Customers.ContactFirstName, T_Customers.Address

T_Customers.Postalcode, T_Customers.City

FROM T_Customers;
```

ACTION QUERIES

An Action Query aims at carrying out modifications in tables or creating new tables.

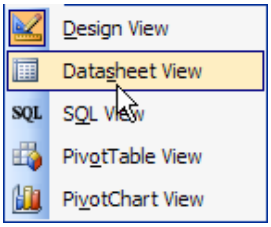
There is an important difference between running the query and previewing the results before carrying it out.

The results can be viewed in a temporary table that is used to check the action. This table always displays the records that will be modified.

Viewing the results

↳ Click the arrow on the right of the  icon.

↳ Choose the ***DATASHEET VIEW*** mode



Running a query

↳ To run a query, click the button  in ***DESIGN VIEW***.

Remark :

You can also select the ***QUERY*** menu and choose ***RUN***.

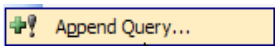
Access asks you to confirm the action. You have to give a positive answer for the action to be carried out.

Once you accepted, **the action cannot be undone anymore**.

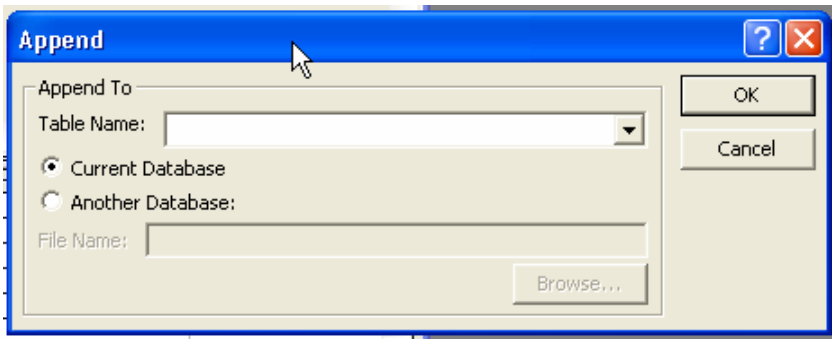
Append query

This type of query makes it possible to add information from a table to another table.

The destination table does not necessarily have to belong to the same database.

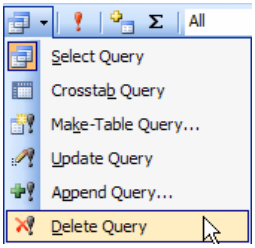
- Make sure that the table you built your query on is indeed the table that contains the data you want to use.
- Click the arrow on the right of the  button;
- Choose the ***APPEND*** option 

in order to specify this type of query.
- Select in the dialog box the destination table where the data will be added. Type the database name if it is another one than the current one.



- Check the name of the field which will contain the information in the row "***APPEND T***" in the grid. Change if necessary.
- Preview the results and run the query.

Delete query

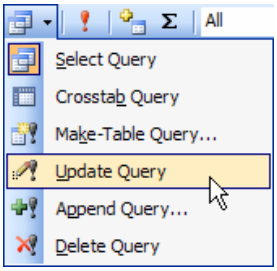


This type of query enables you to delete the data of a table, which respects the desired criteria. Do not forget to specify a criterion; otherwise all the records will be deleted!

Update query

This type of query enables you to modify the records that respect some criteria.

- Choose the update option in the type of queries.



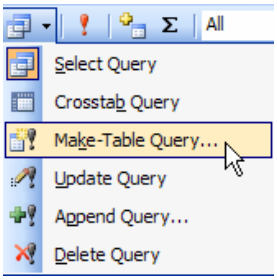
- An additional row appears in the grid for you to enter the new values or expressions that will replace the old information.

Make-Table query

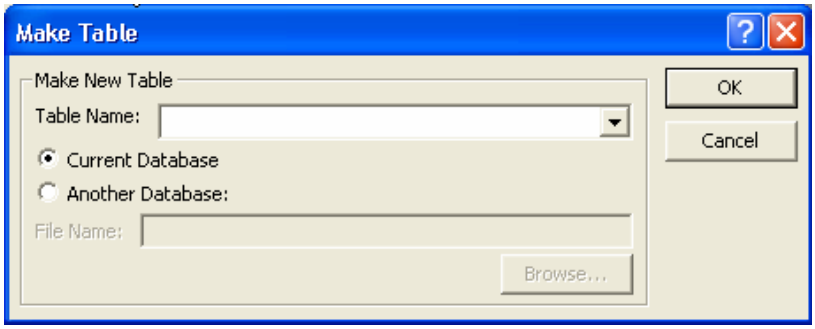
This type of query is used for creating a new table.

In order to change a Select query into a ***MAKE-TABLE*** query:

- Choose the option ***MAKE-TABLE*** query in the type of queries.



- In the dialog box, enter a name for the new table as well as the name of the database if it is another one than the current one.

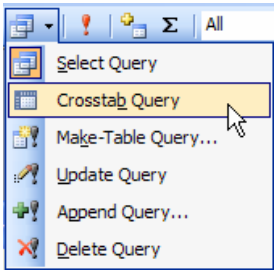


CROSSTAB QUERY

The Crosstab Queries make it possible to group records according to two different pieces of information and present the result as a table with two entries.

> *Example:*
> *You can create a table with the rows corresponding to the products and with the columns corresponding to the years. You also want to display in the cells of this table, the total amount of units sold per product and per year.*

Choose the option CrossTab in the types of queries.



In the grid, enter the fields containing the desired information.

In the ***TOTALS*** row of each field, select the type of action you want to carry out on this field (for example: ***GROUP BY*** for two fields and ***SUM*** for the calculated field).

In the ***CROSSTAB*** row, specify the fields that will make the rows, columns or the fields of which you want to display the calculated values.

This type of query can also be used as the basis for a report.

In the grid, you will find at least three fields, of which two are used for grouping the data and one is used to calculate.

GROUP BY row

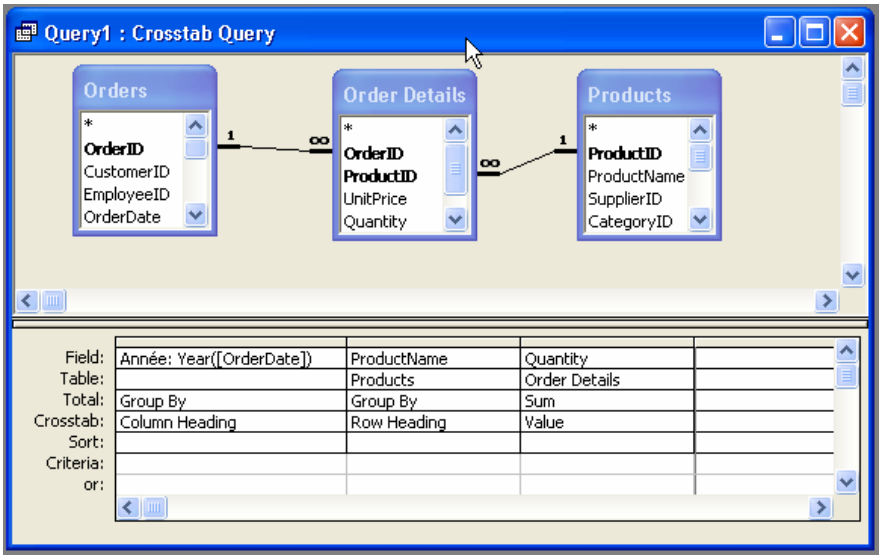
In this type of query, you have to use two different groups (group by) and one field containing a group calculation (sum, min, max, count, avg, etc).

CROSSTAB row

Under the ***TOTALS*** row, the ***CROSSTAB*** row makes it possible to specify the destination of the fields.

You have to define:

- The field that will be used as the ***ROW HEADING*** ;
- The field that will be used as the ***COLUMN HEADING*** ;
- The field that will be used to calculate on (***VALUE***).



Remark :

You can define up to three fields that will be used as row headings

FORMS

In this chapter, you will learn to:

- Create and use a form

A form is a window that allows the user to enter data and view the data contained in tables.

Instead of displaying data in a table, it is possible to use a form that is user-friendlier.

Forms are also used for the development of an application and the creation of menus or dialog boxes.

CONTROLS

The controls are used in a form to make it easy to understand and to use for the final user.

Example :

For a user, there are less risks to enter wrong data by choosing it in a list than by entering it.

Toolbars

A control is an item that will be added in a form, enabling the final user to:

- choose items in a list
- enable or select predefined options
- enter data

To add a control:

Click the button  on the toolbar

The toolbar below appears on screen.

Remark :
All the controls are not available in the forms or reports.

You can create a form by using two methods:

- automatically by using the ***WIZARD***
- manually (more complex but you have more possibilities)

When you use the wizard, make sure the button  is activated.

First make sure you are in ***DESIGN VIEW***

Then click the control on the toolbar

Draw or position the control on the form

Types of controls

There are three types of controls:

- bound
- unbound
- calculated

The ***BOUND CONTROLS*** are controls where the data source is a field of a table or a query. They are used for viewing, modifying the field contents or adding new data.

The ***UNBOUND CONTROLS*** are controls that do not contain any information, neither data from a field in a table, nor a calculation. They are used to display a fixed information, such as a field heading.

The ***CALCULATED CONTROLS*** are the controls whose data source is an expression (calculation, formula, function, …).

The various controls

BUTTON	TYPES	DESCRIPTION
	Select Objects	Used to select objects
	Label	Title
	Text Box	To enter functions, expressions…
	Option group	Used to group various option buttons, check boxes or toggle buttons
	Option Button	Or Radio button, displayed with (if selected) or without a dot inside

BUTTON	TYPES	DESCRIPTION
	Combo Box	Drop-down list where the user will be able to choose data from a hidden source
	Command Button	Push-down button used to: run a macro run a Visual Basic program carry out an action
	Unbound Object Frame	Frame containing an OLE object or a picture with no link to any field of the table
	Page Break	The page break is especially used for reports. It imposes going to the next page
	Line	Simple line of various widths and colors which are used as a separation
	Control Wizards	Allow or not the Control Wizards
	Toggle Button	Button with a binary choice (pushed or not), especially used for pictures and icons
	Check Box	Binary control. The box is checked or not. The symbol "✓" appears in the box if selected, otherwise it is empty
	List Box	List of proposed values. That list box is always opened (>< combo box)
	Picture	Displays a picture
	Bound Object Frame	Frame that contents an OLE object or an incorporated picture, linked to a field of the table
	Subform / Subreport	Displays another report or form inside the current report or form
	Rectangle	Used for the look of the report or to insist on some fields The rectangle can be of different sizes or colors and can be full or empty

THE PROPERTIES WINDOW

All the properties of objects, sections or controls can be modified in this window.

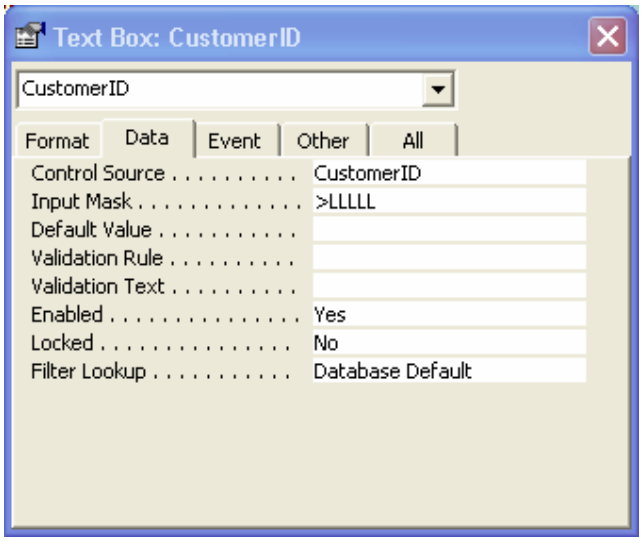
These properties can be classified into five categories :

DATA PROPERTIES:	List with the data properties (default value, format, …)
FORMAT PROPERTIES:	List with the formatting properties (color, font, bold, etc.)
EVENT PROPERTIES:	List with the events with the action linked to each of them
OTHER PROPERTIES:	List with additional properties
ALL PROPERTIES:	List with all the properties

To modify or view the properties of a control :

- Select the control and click the button or click the control with the right mouse button and choose the ***PROPERTIES*** option

- The property dialog box appears.



Each object or element (control) has several properties, which can be modified.

A ***PROPERTY*** describes a characteristic of the object (color, position, …).

Some controls in detail

Label

: use this icon for inserting text, a description, a explanation in a form or a report.

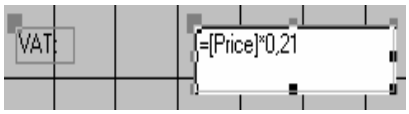
Example

Customer Information

Textbox

: use this icon for calculated fields, functions, input, etc.

Example

IN DESIGN VIEW	*IN FORM VIEW*
	

Option Group

Groups several option buttons, check boxes or toggle buttons.

Example:

An ***OPTION GROUP*** is used mostly with radio buttons and looks like a circle with a dot in it when selected. With each option corresponds a number that will be added– as a number – to the source data field.

By using the ***CONTROL SOURCE*** property of the ***OPTION GROUP*** (the ***DATA*** tab), you can link the control to a field. You can specify the value associated with each option in the ***VALUE*** property of the option.

Check Box

A check box is a status control, presented as a checked or unchecked rectangle. This type of control is typically linked to YES/NO fields.

Example:

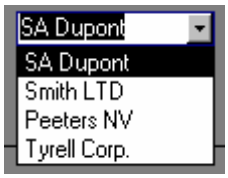
Combo Box and List Box

COMBO: Combines a list box and a text box. You can choose an item in the list or type it yourself.

LIST : list containing data and always displayed as an open list where you can not make any modifications.

Example

In the "clients" field of the "commandes" form, a list with the customers is used instead of a single text box where the user would have to enter the customer's code.



These are the useful properties for the pick lists:

In the ***DATA*** tab:

ROW SOURCE TYPE: this option enables you to specify the type of pick list. ***TABLE/QUERY*** makes it possible to create a list of which the values come from a table or a query, and ***VALUE LIST*** enables you to create a list where you will enter the values for the list.

ROW SOURCE: If you have chosen the option ***ROW SOURCE TYPE=TABLE/QUERY***, you can either choose an existing table or query, or type the SQL code of a query. To make it easier, you can also click the button "…" which gives access to the query generator. If you have chosen the option ***ROW SOURCE TYPE=VALUE LIST***, you can enter here the values you want the list to contain, separated by ";".

BOUND COLUMN: In case the table or the query that was used to create the pick list contains several fields, you have to specify which field will really be used. To do so, you have to specify here the number of the chosen column ; the left column being considered as the column number 1.

In the ***FORMAT*** tab:

COLUMN COUNT: Indicate here the number of columns for the pick list. These columns go from left to right in the data source that is mentioned in the ***ROW SOURCE*** property.

COLUMN HEAD: If you choose ***YES*** in this option, the pick list will display a column heading.

COLUMN WIDTH: This property enables you to specify the column width. If the pick list contains more than one column, you will have to separate the column widths by using ";". If you want to hide a column, simply enter nil for the width.

The pick lists based on existing tables often have the same structure: the data source is a query containing the key of the table as well as a description field, alphabetically sorted on the description. The linked column is the one with the key. In the ***FORMAT*** tab, the number of columns is fixed : 2, but the first one is hidden (Column Width=0;5).

REPORTS

In this chapter, you will learn to:

- Create and print a report

You can add controls in reports, as in forms.

A report is not interactive, this means that you cannot use Command buttons, Combo Boxes, etc…

The reports (as the forms) contain several sections.

SECTIONS

Detail (always visible)

For the reports, this section is repeated for each record of the query or the table that was used to base the ***REPORT*** on.

Form/Report header and footer

For the reports, the information of this section is only printed on the first or the last page.

Page header and footer

For the reports, the information of this section is printed at the top (title, page number) or at the bottom (totals) of all the pages of the report.

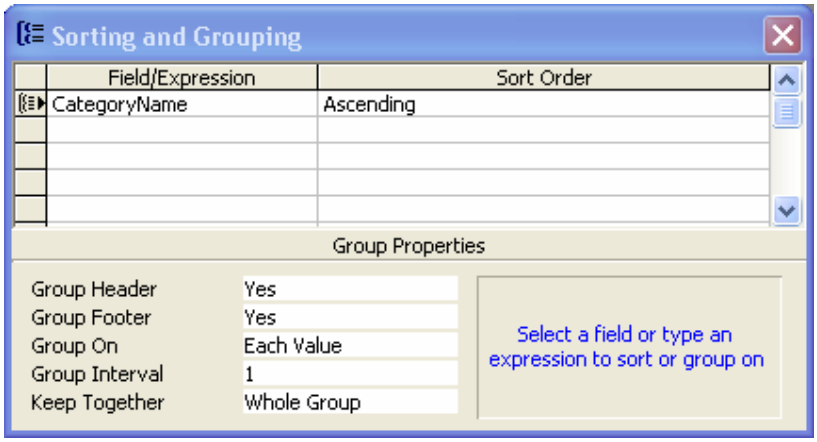
The controls

The controls are all the objects that can be added. The sections have their own properties, the report also.

SORTING AND GROUPING

To display the window ***SORTING AND GROUPING :***

- Click the button  or choose the command ***VIEW.***
- Choose ***SORTING AND GROUPING.***



Sorting

It is possible to display the information in a different order than the one in the table.

- In the ***FIELD/EXPRESSION*** area, select the field you want to sort on.
- Choose ascending or descending in the ***SORT ORDER*** area.

Grouping

It is possible to group the records containing the same value in a field.

> *Example:*
>
> *You can print the customers' reports by region, or by product category.*

- Activate the ***SORTING AND GROUPING*** window.
- In the ***FIELD/EXPRESSION*** area, choose the column containing the records you want to group.
- Choose a sort order in the ***SORT ORDER*** area (ascending or descending).
- If you want to, you can add a header or footer for the group (***GROUP HEADER OR GROUP FOOTER***).

These are the parameters for regrouping:

GROUP HEADER: adds a group header

GROUP FOOTER: adds a group footer

GROUP ON: as a standard, the groups are based on individual values (***EACH VALUE***). For the text fields, you can group the records by using the first letters of the field as the criteria for grouping (***PREFIX CHARACTERS***). For example, you can group the customers by displaying all the customers whose name begin with an A,

then those whose name begin with a B etc. When grouping numerical fields, you can choose to group per intervals (***INTERVAL***).

Group Interval: if you have not chosen the option ***GROUP ON=EACH VALUE***, you can specify here the used interval. For the text fields, you can specify the amount of characters that were taken into account. For the numerical fields, you can specify the interval that was used for the grouping.

KEEP TOGETHER: If you choose the option ***WHOLE GROUP***, you will obtain a report where the groups are, if possible, kept entirely on the same page. If you choose the option ***WITH FIRST DETAIL***, there will never be any page break between the group header and the first record of the group.

THE MACROS

In this chapter, you will learn to:

- Create and run a macro

WHAT IS A MACRO?

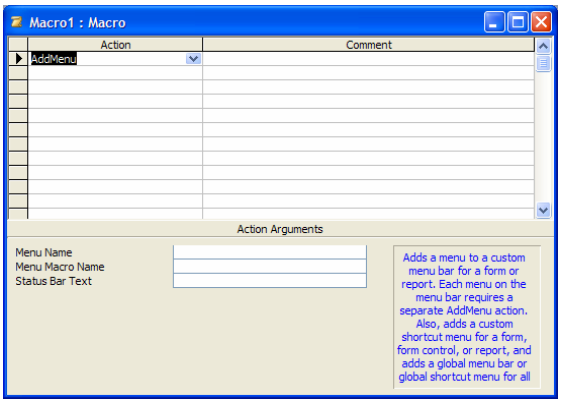
A Macro is a unique action or a series of several actions stored as a database object. You can run it whenever you need it.

Macros are important for automating and simplifying the use of tables, queries, forms and reports. You can automate all the tasks with a Macro and run them with a simple click.

The ACCESS Macros do not need code" programming. It is in the modules that we will be able to make small programs in Visual Basic

CREATING A MACRO

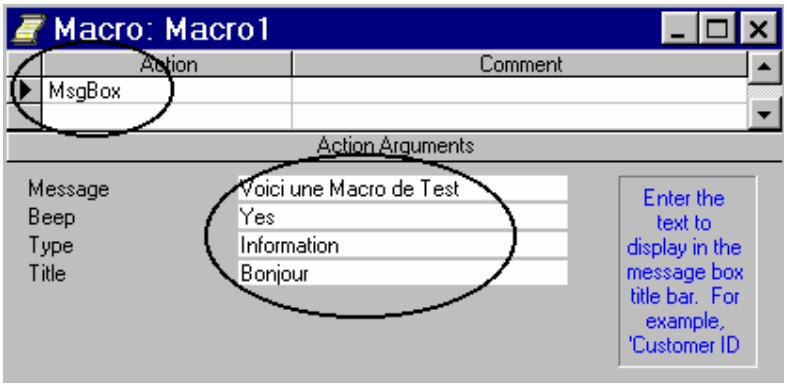
In order to create a Macro, click the ***MACROS*** tab in the database window and then click the ***NEW*** button. A new Macro sheet is displayed :



The first column " Action " contains the list with all the actions of the macro. If you select an action, the arguments (the properties) of this action will be displayed at the bottom of the window.

The parameters can be entered as text or can be selected in a list with options.

The second column is an optional column, for a description. It is only useful to keep some notes for yourself or for any person who might need to understand or modify the macro later on.

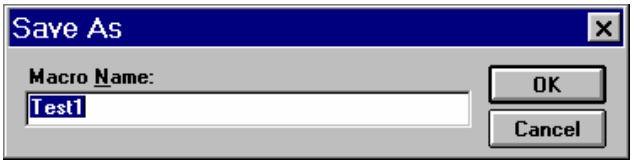


For example, this macro with the above mentioned parameters will give the following message.

The ***MESSAGE*** argument contains the contents of the message, ***BEEP*** is the sound, ***TYPE***, the icon with the small "i " and ***TITLE*****,** contains the title of the titel bar.

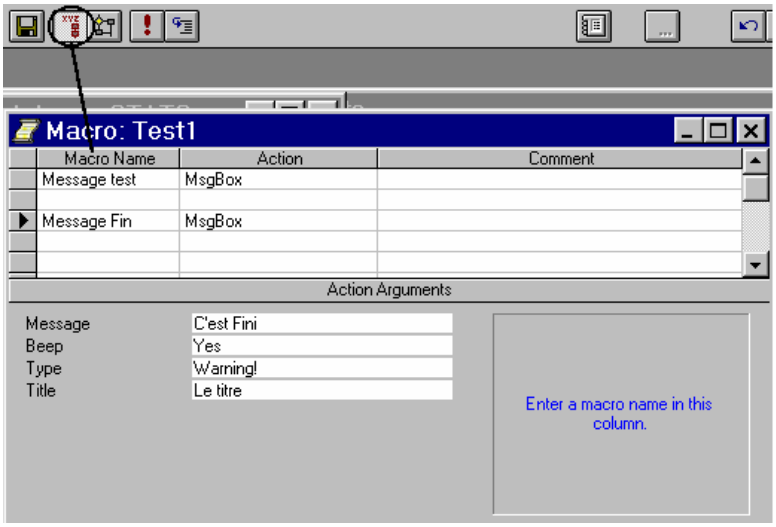
NAMING MACROS

Once you created a macro, you have to name the macro before being able to run it. As for the other objects; you will enter the name in a small dialog box.



You will probably create more and more macros later on. But do not create one sheet per Macro.

So you can create several Macros per sheet. It will then be necessary to display an additional column in your sheet: the Macro Name column by clicking the " xyz " button.



This window contains 2 Macros : ***MESSAGE TEST*** and ***MESSAGE FIN*** in the same sheet Test1.

The Macro will then be named according to the following syntax: Name of the sheet**.**Name of the Macro

Test1.Message test

Remark :

Be careful: from this moment, you will not be able to run the macros by using the Macro sheet (only the first one). You will have to use the ***EVENTS*** to run the other ones.

RUNNING A MACRO

If you want the macro to really carry out the actions, you have to run the macro.

There exist several methods or situations you can use to start the macro.

Starting the macro in the Database window

- Click the ***MACRO*** tab.
- Double-click the name of the macro you want to run or select its name and click ***RUN***.

Starting the macro in the Macros window

- Click the ***RUN*** button.

ORGANIZING AND MODIFYING MACROS

A macro can contain more than one action.

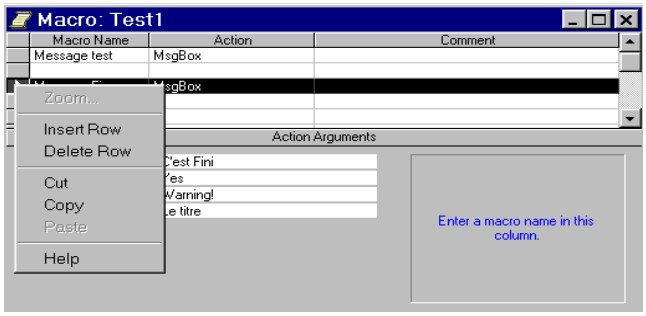
So the actions will be selected row per row.

ACCESS will carry out the commands in the order you placed them in. This means that it will first carry out the first one (the top one), then the following action and so on until the last action in the Macro.

But it is possible to delete, insert or move actions in the macros.

Deleting an action

- Select the row containing the action (by clicking the row selector) you want to delete.
- Then press the ***DELETE*** key or right-click and choose ***DELETE ROW.***



Inserting an action

- Select the place where you want to add your action.
- Then press the ***INSERT*** key or right-click and choose ***INSERT ROW.***

Moving an action

- Click the row selector.
- Hold down the mouse button and drag the row(s) to its new place.

The actions in the macros

The following table contains the various actions available in the Macros with a short explanation. If you want more details about these various possibilities, you can have a look at the help.

CATEGORY	ACTION	TASK
Data in forms and reports	ApplyFilter	Restricts the data by applying a filter
	FindRecord, FindNext	Is the same as the Find command
	GoToControl, GoToPage, GoToRecord	Makes it possible to go to a control, a page or a record specified in the arguments
Running	DoMenuItem	Runs a command from the menu
	Quit	Quitting Microsoft Access
	OpenQuery, RunCode RunMacro RunSQL	Runs a Query a procedure a macro a Query in SQL language
	RunApp	Runs another application
	CancelEvent StopAllMacros StopMacro	Stops the running all the macros the running macro
Importing/exporting	OutputTo, SendObject	Sending Microsoft Access objects to other applications
	TransferDatabase, TransferSpreadsheet, TransferText	Transferring data between Microsoft Access and other data formats such as Excel or Word
Manipulating objects	CopyObject, Rename	Copying or renaming an object
	DeleteObject	Deleting an object
	Maximize Minimize MoveSize Restore	Moving or changing the size of a window

CATEGORY	ACTION	TASK
	Close OpenForm OpenModule OpenQuery OpenReport OpenTable	Opens or closes an object
	Print	Prints an object
	SelectObject	Selects an object
	SetValue	Sets the value of a field, a control or a property
	RepaintObject Requery ShowAllRecord	Updates the data or the screen
Other	AddMenu	Creates a menu bar, a customized contextual menu, a global menu bar or a global contextual menu
	Echo Hourglass MsgBox SetWarnings	Displays the information on the screen Displays the hourglass during an action Displays a customized Message Box Displays or not warning messages
	SendKeys	Generates a keystroke
	ShowToolbar	Displays or hides the integrated or customized toolbar
	Beep	Makes a sound (Beep !)

The arguments

After having added an action to a macro, specify the arguments of the action in the lower part of the macro window.

These arguments give ACCESS additional information about how to run the action.

These are a few pieces of advice on how to define the arguments of the action :

- You can enter a value in an argument box or you can often select a parameter in a list.
- As a general rule, it is advised, when defining the arguments of the actions, to respect the order of the list, as the choices of an argument can determine those of the next arguments.

- If you add an action to your macro by dragging an object from the Database window, ACCESS will automatically define the right arguments for this action.
- If the argument of an action calls for the name of an object, you will be able to define automatically the argument as well the argument Object Type by dragging the object from the Database window to the argument box.

THE CONDITIONS IN MACROS

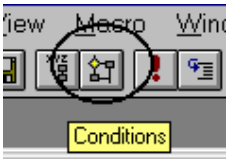
In some cases, you might want to run an action or a series of actions only if a specific condition is true. For example, if you use a macro to validate the data of a form, you might want to display a message in response to a group of values entered in a record and another message in response to another group of values. In this case, you can use conditions in the macro.

A condition is a logical expression. The macro follows different paths whether the condition is true or false.

Viewing the column with the conditions

To display the column with the conditions :

Click the button.



Enter the conditions in the column " Condition " in the Macro window.

If a condition is true, ACCESS will run the action in the same row.

You can ask ACCESS to run a series of actions if the condition is true by entering 3 consecutive dots (...) in the column " Condition " of the actions that follow immediately the condition.

Example:

I would like to display an American flag in the Form " Fournisseurs" when I display the record of an American supplier. By using a macro, I will display or hide the American flag by changing the property Visible.

Let us start by adding in the ***FORM*** the picture of a flag and name this control USFLA.

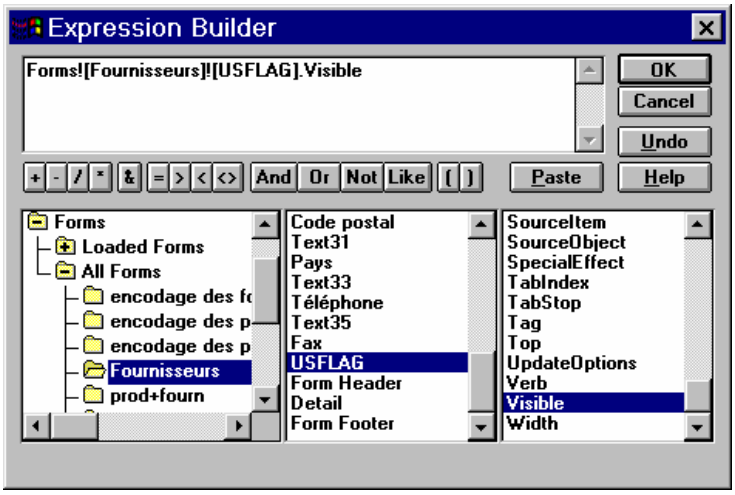
The standard value of the property Visible is ***YE***s.

The aim of the macro is to display ***NO*** in the property Visible if it is not an American supplier. Use the ***SETVALUE*** action***.***

The parameters are

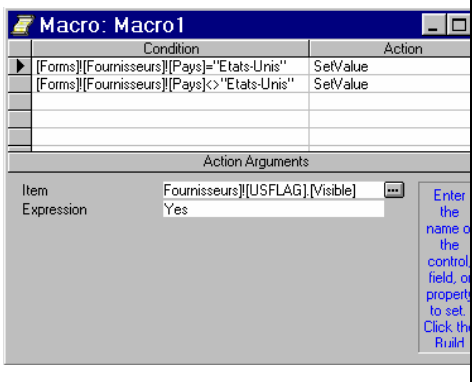
Item	Control, Field or Property to define. Here, the property Visible of the control USFLAG in the “Fournisseurs” Form is: Forms![Fournisseurs]![USFLAG].Visible
Expression	Value for the object. Yes or No

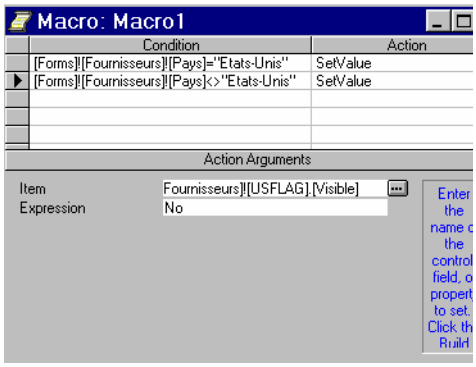
* To create this expression, you can use the ***EXPRESSION BUILDER*** by clicking the ***...***

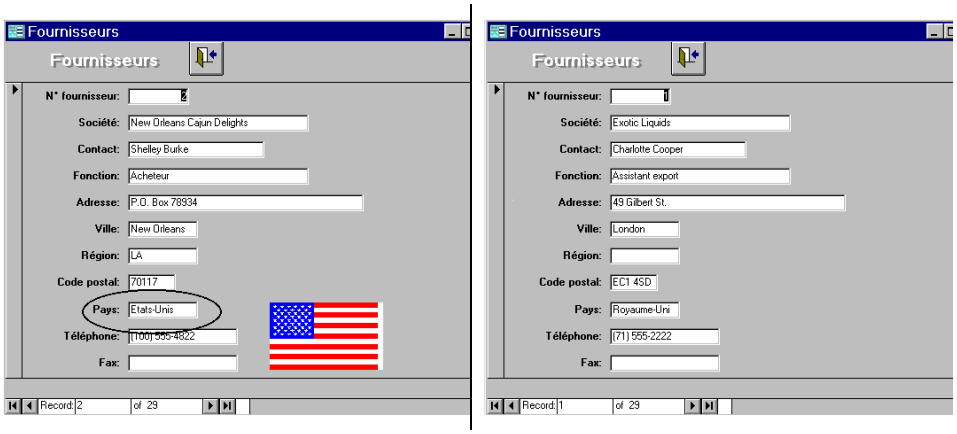


We still have to define the conditions

American supplier	Non-American supplier
Visible flag: YES	Visible flag: NO
Condition [Forms]![Fournisseurs]![Pays]="Etats-Unis"	Condition [Forms]![Fournisseurs]![Pays]<>"Etats-Unis"







Examples of conditions in a Macro

USE THIS EXPRESSION	*TO RUN THE ACTION IF*
[Ville]="Paris"	The value « Ville » in the field of the Form for executing the macro is Paris
DCount("[N° commande]", "Commandes")>35	There exist more than 35 entries in the field “N°commande” of the table “Commandes”
DCount("*", " Détails Commandes ", "[N°commande]=Forms![Commandes]![N°commande]")>3	There exist more than 3 entries in the table “Détails Commandes” where the field “N°commande” of the table corresponds to the field “N°commande” of the Form “Commandes”
[Date d'envoi] between #2-Fév-1995# et #2-Mar-1995#	The value of the field « Date d'envoi » of the Form for executing the macro ranges between « le 2-Fév-1995 » and « le 2-Mar-1995 »
Forms![Produits]![Unités en stock]<5	The value of the field « Unités en stock » of the Form « Produits » est inférieure à 5
IsNull([Prénom])	The value « Prénom » in the Form for executing the macro is Nil (non-existent). This expression is the same as « [Prénom] IsNull »
[Pays]="Royaume-Uni" And Forms![Total de ventes]![Commandes cumulées]>100	The value of the field « Pays » in the Form for executing the macro is « Royaume-Uni », and the value of the field « Commandes cumulées » in the Form « Total des ventes » is bigger than 100

[Pays] In ("France", "Italie", "Espagne") And Len([CodePostal])<>5	The value of the field « Pays » in the Form for executing the macro is « France », « Italie » or « Espagne » and the postal code is maximum 5 characters
MsgBox ("Confirmer changements?" ,1) = 1	You click OK in a dialog box displaying by using the function MsgBox. If you click Cancel in the dialog box, Access will ignore the action

Remark :

When there are at least two conditions, it is useful to use the action StopMacro at the end of each condition in order to avoid having to look at the whole macro.

So, if the first condition is " true ", the actions will be carried out and you can use StopMacro to stop the macro. Access will nevertheless read the second condition and its actions.