Access Visual Basic

DoCmd object

allows you to run a macro command within a VB module

DoCmd object

Many of the action methods of the DoCmd object have parameters or arguments (required or optional).

DoCmd object don't work for the following macro actions:

- **AddMenu**
- **MsgBox** - Replaces by MsgBox function
- **RunApp** - Use the Shell function
- **RunCode** - Run the function directly in VBA
- **SendKeys** - Use the SendKeys command
- **SetValue** - Assign values directly in VBA
- **StopAllMacros**
- **StopMacro**

DoCmd.RunSQL

Access Database Example

The Access **RunSQL method** performs the RunSQL action in VB.

This command is used to execute SQL query code within Access VB.

MS Access RunSQL method

SQL Statement - A required variant string expression that is a valid SQL statement for an action query or a data-definition query. Examples are **Update**, **Delete**, **Insert Into**, etc., queries.

Access RunSQL Example:

```
Public Sub RUN_Query
    Dim  txSQL as String
    txSQL = "DELETE * FROM  Employees"
    Docmd.RunSQL (txSQL, false)
End Sub
```

Use False (0) if you don't want to use a transaction.

The RunSQL command

is one of the most powerful features of Access VB.

The programmer has:

- the ability to perform all the logic & data validation functions within VBA *and*
- has the power of SQL.

DoCmd Run SQL	Run an SQL action query by using the SQL command. You may also run a DDL query

The following list of **DoCmd methods** shows the commonly used commands. The general format for the command is

DoCmd.*Action*. There are often 1+ options to these action methods

DoCmd Add Menu	The AddMenu command example demonstrates how to add an item to customized dropdown menus at the top of the MS Access database screen.
DoCmd Apply Filter	These examples employ the Docmd.ApplyFilter method to display only records that contain the state MD in the State field.
DoCmd Cancel Event	The *Cancel Event* method quits the event that caused Access to run the code containing this action. All further execution of the event is aborted.
DoCmd Close	The Access Close method carries out the Close action in VB to **close a form, report, or query** in the running database.
DoCmd Copy Object	Docmd.CopyObject allows you to **duplicate any database object**.
DoCmd Delete Object	The DoCmd DeleteObject method performs a **delete operation on Access's objects such as tables, queries, forms and reports**.
DoCmd Find Record	The Access DoCmd.FindRecord method performs the Find Record action in VB. This command is often used in conjunction with the Docmd.FindNext to automate searching for datasheet records based on user input.
DoCmd Find Next	Find the next record that meets the criteria specified by the previous FindRecord action or the Find In Field dialog box.

List of DoCmd methods shows the commonly used commands (2):

DoCmd Goto Control	Move the focus to the indicated field or control in the current record of the open form, datasheet, table datasheet, or query datasheet.
DoCmd Hour Glass	Change the mouse pointer to an hourglass icon (or another icon of your choice) while program is running. This method is used when running a procedure from VBA that takes a while to run.
DoCmd Minimize	Minimize (+maximize) docmd shrinks the active window down to the smallest size available while still showing the forms close and min/max size icons (/max dim) .
DoCmd Restore	Docmd Restore returns the size of the active form to the setting previous to a maximize or minimize docmd execution.
DoCmd Open Form	Open in Form view, datasheet, print preview, or design view
DoCmd Open Query	Open a select or crosstab query in design, datasheet, or print preview. Or, execute an action query from VBA script
DoCmd Open Report	Open a report in design or print preview, or to print the report directly to the default printer.
DoCmd Output To	Output data in the current database (a datasheet, form, report, module, DAP) to a file in Excel, DOS text (*.txt), or rich-text format.
DoCmd Run SQL	Run an SQL action query by using the SQL command. You may also run a DDL query
DoCmd Send Object	Include the specified datasheet, form, report, module, or data access page in an e-mail message - there it can be viewed and sent.
DoCmd Transfer Text	The TransferText method sends/receives records from/to the db or project & a file on your hard drive. The transfertext method may also be employed to establish data table links between DBs.
TransferDatabase	**Import or export data from/to the current database** (.mdb) or project (.adp) and other databases such as FoxPro, dBase, and ODBC databases.

Examples of SQL being implemented from VBA

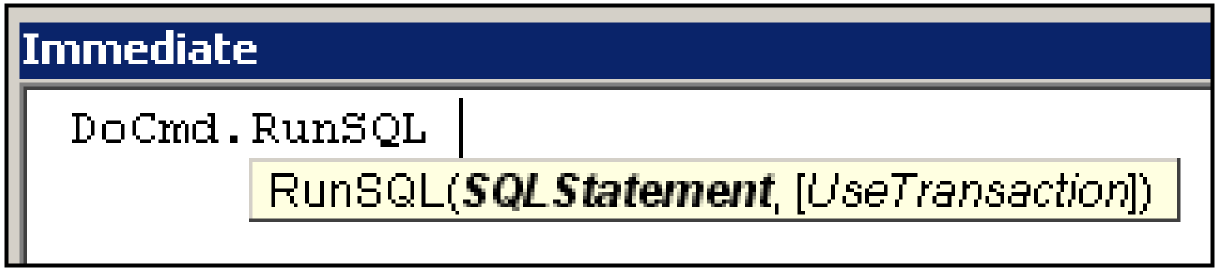
The *DoCmd* object in VBA can be used to perform a wealth of different actions including one called *RunSQL*.

You can't run *any* sort of SQL statement using the *RunSQL* method, it is specifically for running the type of queries that Access calls "action queries":

These include:	used to:
• *Delete Queries*	(**delete records** from a table)
• *Append Queries*	(**add records to** a table)
• *Update Queries*	(**edit records** in a table)
• *Make Table Queries*	(**create a new table**).

Working from the Access query design window you are limited to the four query types described above, but using SQL in conjunction with VBA (or by entering SQL directly into the SQL view of the Access query design window) you can accomplish a lot more, including the use of "data-definition queries" which are used to build and modify the structure of the database itself.

The *RunSQL* method prompts you for two arguments, the **SQL Statement** itself which must be supplied as a string (i.e. it should be enclosed in quotes) & **Use Transaction** which is optional:



```
DoCmd.RunSQL
RunSQL(SQLStatement, [UseTransaction])
```

Build a New Table

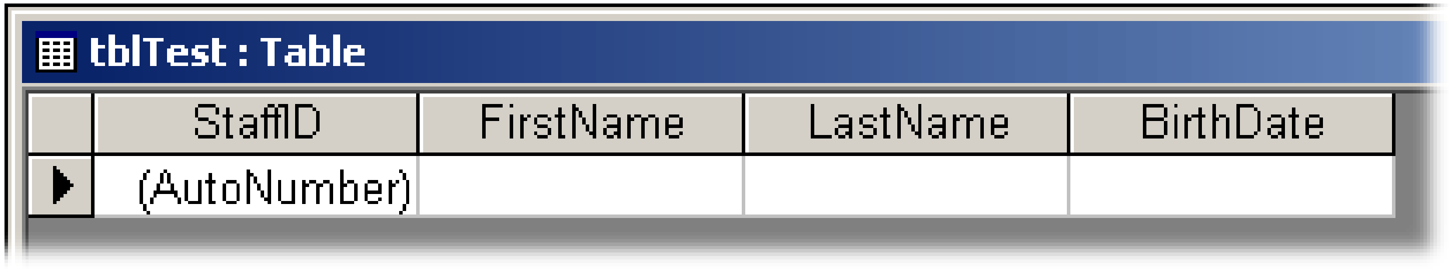
Open a VBA code window (**Alt+F11** > **Modules** tab & click **New**).

For the purpose of this exercise, you will run the code directly from the **Immediate Window**. This window allows you to implement a code statement directly by typing in into the window (it has to be typed as a single line) and pressing **Enter**.

Open the *Immediate Window* **(Ctrl+G)**. Enter the following line of code as a single line, then press **Enter**:

```
DoCmd.RunSQL "CREATE TABLE tblTest (
        [StaffID] COUNTER CONSTRAINT ndxStaffID PRIMARY KEY,
        [FirstName] TEXT(25),
        [LastName] TEXT(30),
        [BirthDate] DATETIME);"
```

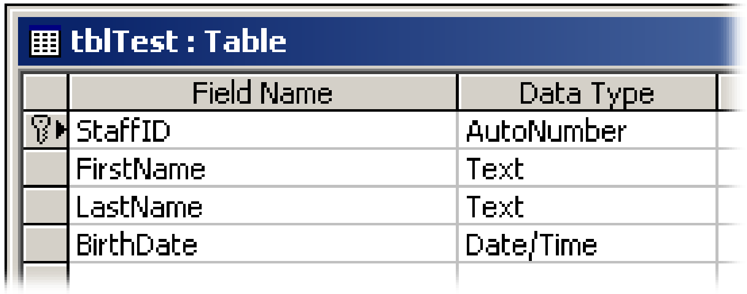
Switch to the Access DB win (**Alt+F11**) and move to the **Tables** tab. You should see that a new table (*tblTest*) has been created. Take a look at the table. You will see that it contains the four fields specified in the SQL statement:



tblTest : Table

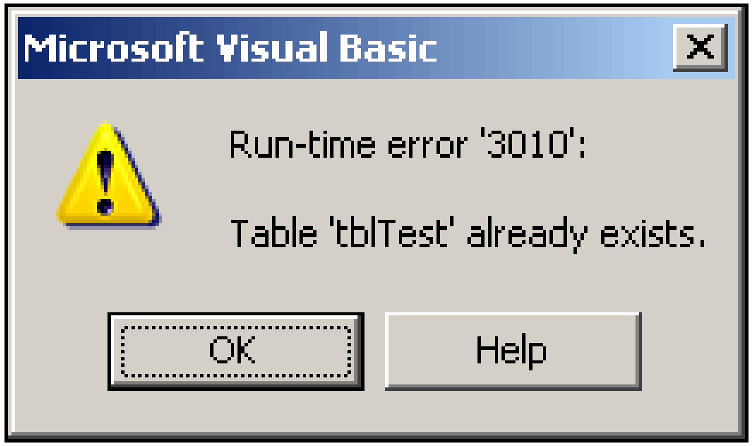
	StaffID	FirstName	LastName	BirthDate
▶	(AutoNumber)			

Switch it into design view and see that the data types are as specified, with the text field of a specified size, and that the *StaffID* field is an autonumber field and also the *primary key* field:


tblTest : Table

Field Name	Data Type
StaffID	AutoNumber
FirstName	Text
LastName	Text
BirthDate	Date/Time

If you try to run the same line of code again an error occurs because a table with the specified name already exists:



Run-time error '3010':

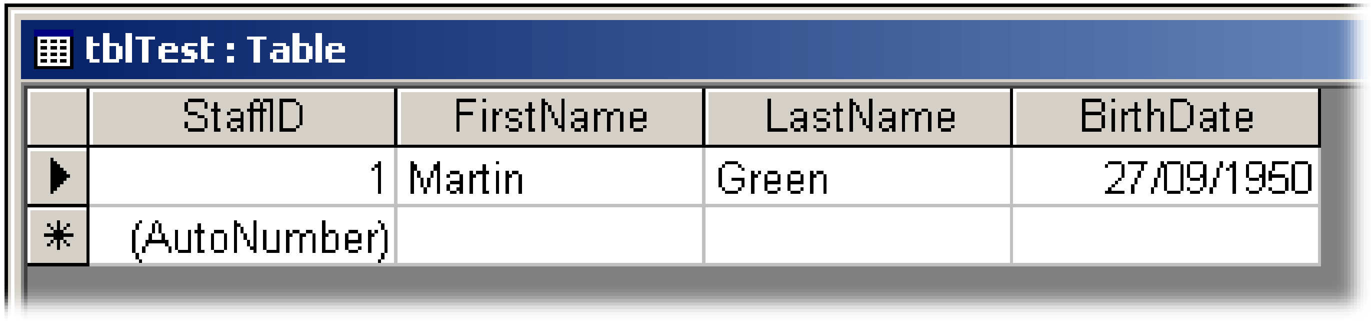
Table 'tblTest' already exists.

Add Records to a Table

Lets add some data to the table. **Close** the table if it is open, & return to the *Immediate Win*. Enter the following line of code as a **single line**, then press **Enter** :

```
DoCmd.RunSQL("INSERT INTO tblTest (
    [FirstName],  [LastName], [BirthDate]) VALUES
    ('Martin', 'Green', #09/27/1950#);") )
```

Note the single quote marks around the text values *Martin* and *Green*, and remember that *single* quotes are used here so as not to conflict with the double quotes that enclose the complete SQL statement. Note also that the date value is enclosed by hash marks (***#***) and that the date is supplied in US (m/d/y) format. Switch to the Access DB Win & open the table. You will see your new record. Because the *StaffID* field is an *autonumber* field its value is assigned automatically. It does not need to be specified in the SQL.

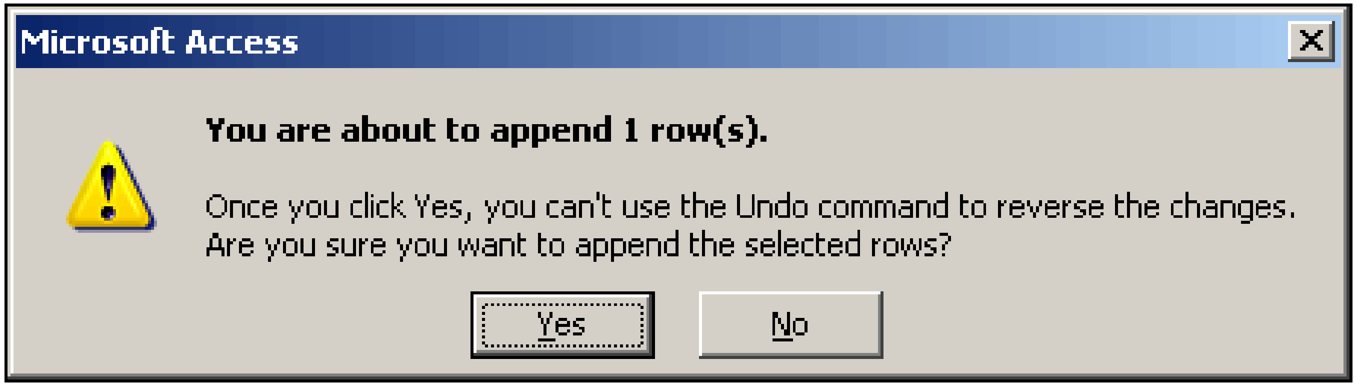


tblTest : Table

	StaffID	FirstName	LastName	BirthDate
▶	1	Martin	Green	27/09/1950
*	(AutoNumber)			

Add Records to a Table

Before the SQL is executed, Access displays a message asking permission to add a record to the table. This is usual when Access performs any action query, and can be suppressed with VBA code if you don't want your users to see it:

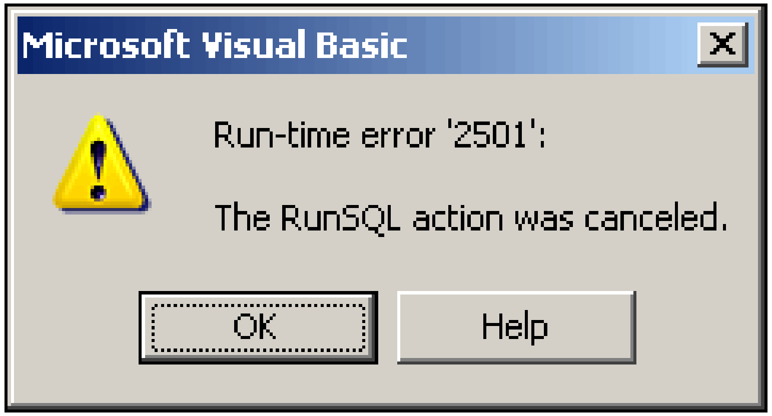


Microsoft Access

You are about to append 1 row(s).

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

If the user decides not to append the record they can click the **No** button and the action is cancelled without any further consequences, but when this happens when the SQL statement is being run from VBA an error occurs:



Microsoft Visual Basic

Run-time error '2501':

The RunSQL action was canceled.

Add a Field to a Table

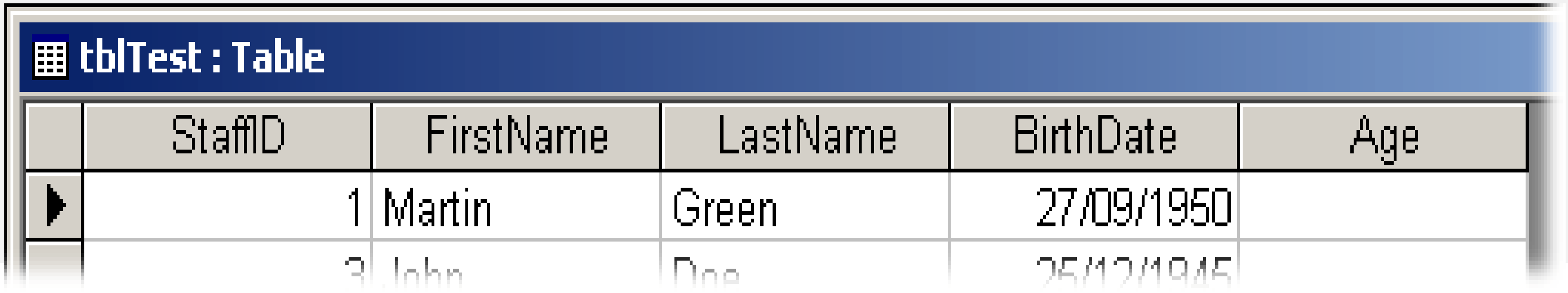
SQL can be used to make changes to the structure of an existing table. Fields can be added, removed or changed.

Close the table (if it is open) & go to the *Immediate Win*.

Enter the following line of code as a single line:

```
DoCmd.RunSQL("ALTER TABLE tblTest ADD COLUMN Age BYTE;")
```

Switch to the Access db win & open the table. You will see:



tblTest : Table

	StaffID	FirstName	LastName	BirthDate	Age
▶	1	Martin	Green	27/09/1950	
	2	John	Doe	25/12/1945	

Modify Existing Records

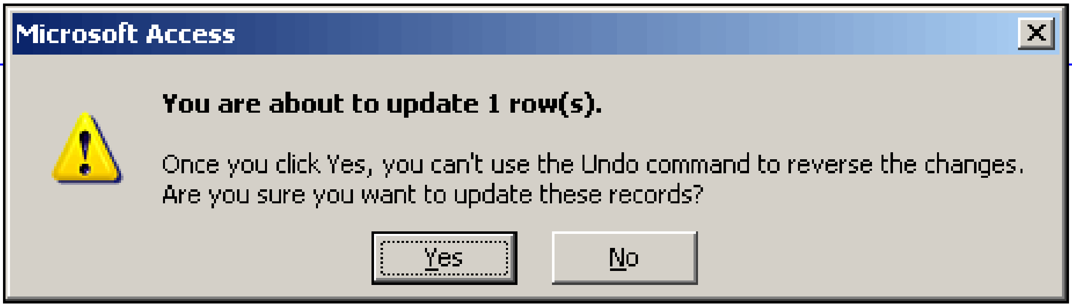
In addition to working with the structure of a table, **SQL can be used to modify existing data**. You may have used an Access Update Query. This example uses the same method from VBA. We have added a new field to the table, now we can enter some data.

An example of updating specific records by adding a WHERE clause to the SQL statement:

Enter the following line of code as single line into the *Immediate Win:*

```
DoCmd.RunSQL  _
"UPDATE Tbl SET Age=52  WHERE FName=’Joe'  AND Name='Gren’;”
```

As when you added records to the table, Access displays a confirmation message when records are about to be updated. Remember that cancelling the update will raise an error in VBA.



Microsoft Access

You are about to update 1 row(s).

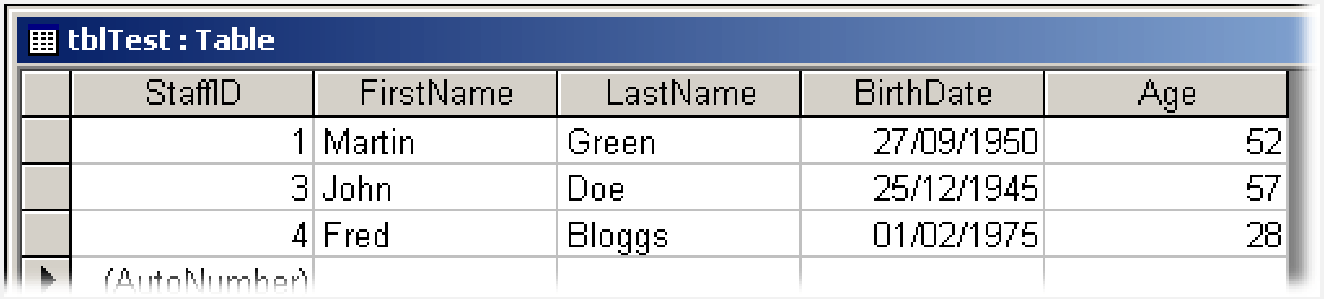
Once you click Yes, you can't use the Undo command to reverse the changes.
Are you sure you want to update these records?

Modify Existing Records (2)

Enter the following line of code as a single line into the *Immediate Win*:

```
DoCmd.RunSQL "UPDATE tblTest
    SET Age=Int((Date()-BirthDate)/365.25); "
```

Instead of applying a specific value, this SQL statement performs a calculation on each record making use of the value already present in the *BirthDate* field and the Access *Date()* function to calculate each person's age:



tblTest : Table

StaffID	FirstName	LastName	BirthDate	Age
1	Martin	Green	27/09/1950	52
3	John	Doe	25/12/1945	57
4	Fred	Bloggs	01/02/1975	28
(AutoNumber)				

NOTE: It isn't good database practice to store calculated data in a table that also contains the data from which it was calculated.

Why? Mainly because it wastes space. If you know a person's birth date you can calculate their age at any time using a query. Also, if you store their age as a number it will not update itself as time passes, so eventually it will become incorrect. But I was stuck for an idea.

Delete a Table

Records can be deleted, as can fields and even entire tables:

Enter the following line of code as a single line:

```
DoCmd.RunSQL "DROP TABLE tblTest;"
```

You won't see any warning message, but you will find that the table has gone (a bit too easy for comfort!).

Summary

These practical examples have demonstrated how you can manipulate a database's structure and its data by implementing SQL statements with VBA, working independently of the Access query tool.

They show the potential of working directly with SQL from your VBA procedures to build, modify and populate tables with ease.

SQL is capable of a great deal more.