


Database access from HBasic programs

Contents

- [Accessing a table from a HBasic program](#)
- [Accessing all records of a table](#)
- [Positioning the edit cursor](#)
- [Changing existing data](#)
- [Inserting new data](#)
- [Deleting records from a table](#)

Prerequisites

 To use database access you need to install QT in a version higher than 3.0 and compile the database plugging matching the kind of your database server. Remember to include the package `hbasic_dbaccess` if you create new database examples. If you load the predefined examples from the HBasic distribution this package will be loaded automatically.

Accessing a table from a HBasic program

Before you may use the methods described below you always have to connect to a database table. This connection will be set up with two components from the package `hbasic_dbaccess`. The first component called `dbconnection` creates the database connection and the second component called `recordset` defines an edit buffer that may be used to display or change the records of a database table.

Preparing a database connection normally looks like the following code segment.

```
Dim dbc As dbconnection
Dim t As recordset

dbc.connect( "hbasic", "root", "orange" )
t.open( dbc, "tab1" )
```

First an instance of the two components will be created with the Dim statement. In HBasic predefined components will be created without a *New* statement. The **database connection** will be initialised with a call to the method `connect` of the `dbconnection` component. The parameters are the name of the database you want to connect to, the user that should be used and the password for the user. As you can see in the example my test database is called `hbasic` and I connect with the user `root` and the password `orange`. Replace this names with the name and user of your database if you want to start the examples

The **recordset** that will be used for further data changes will be initialised with a call of the `open` method of the `recordset` component. The first parameter is the database connection and the second parameter is the name of the table that should be used in the recordset.

You will find this lines always on the first lines of the following source code examples.

Reading all records of a table

This examples sets up a loop which will step through all rows of a table `tab1` and show the value of the column `col1` for each dataset.

The select example selects all rows from the table `tab1` and prints the value of column `col1` for each row.

```
Dim v As Variant

Sub button1_clicked()
    Dim dbc As dbconnection
    Dim rs As recordset

    dbc.connect( "hbasic", "root", "orange" )
    rs.open( dbc, "tab1" )

    While Not( rs.eof() ) Do
        v = rs.getvalue( "col1" )
        Print v

        rs.movenext()
    Wend
End Sub
```

Example `ex_db_read.bas`: Read the value of `col1` from all table columns in `tab1`

Positioning the edit cursor

If you want to change data within a table of your database you have to set the current position of the edit cursor to the depending row first. This may be done by using one or more of the following commands.

Command	Description
MoveNext	Move edit cursor to next record of table
MovePrev	Move edit cursor to previous record of table
MoveFirst	Move edit cursor to first record of table
MoveLast	Move edit cursor to last record of table
Seek(position_n)	Set edit cursor to position_n in table

After each of this operations you may test if HBasic has found a valid record with the function EOF(). If HBasic cannot position on a valid record the function EOF will give a result of TRUE. You may for example loop through all records of a table with the following While command:

```
While Not t.eof()  
  Print t.GetValue( "col1" )  
  t.MoveNext()  
Wend
```

Other methods for database components

The package *hbasic_dbaccess* provides the following components and methods.

Component db_connection

connect(db_name, user_name, user_password) Create new connection to database <db_name>

Component recordset

open(db_connection, table_name)	Open recordset for table <i>table_name</i>
bool eof()	return TRUE if no more recordsets can be read
delete()	Delete current row from table
update()	Store values changed with setValue to database
addNew()	Add new row to table and make it current row
variant getValue(column_name)	Read the value of column_name and return as variant
setValue(column_name, new_value)	Change value of column_name to new_value

Update current row in database

You may change the data in the current database record with the method setValue(<columnname>, new_value). This method will only change the data in the current edit buffer in memory. Call the method *update* to store the changed data in the database.

```
t.setValue( "col1", 123 )  
t.Update()
```

Change values of the current row in table tab1. Since we do not position the row-pointer we change the first row in the table.

```
Dim v As Variant  
  
Sub button1_clicked()  
  Dim dbc As dbconnection  
  Dim rs As recordset  
  
  dbc.connect( "hbasic", "root", "" )  
  rs.open( dbc, "tab1" )  
  v = 1111  
  
  rs.setValue( "col1", v )  
  rs.update()  
End Sub
```

Example *ex_db_update.bas*: Update the value of column col1 in table tab1

Insert new database row

You may insert a new record into a table represented by a recordset. Add a new record to the table with the method *addNew()*. This will create a new record in the edit buffer. You can now change the values of the edit buffer with the *setValue(<columnname>, new_value)* method. As with the update method you have to call the method *update* to store the changed data in the database.

```
t.addNew()  
t.setValue( "col1", 3333 )  
t.Update()
```

Insert a new row into table tab1 and set values for new row.

```
Dim v As Variant  
Dim pos As Long  
  
Sub button1_clicked()  
    Dim dbc As dbconnection  
    Dim rs As recordset  
  
    dbc.connect( "hbasic", "root", "" )  
    rs.open( dbc, "tab1" )  
  
    v = 1111  
    pos = 1  
    While pos <= 10  
        rs.addnew()  
        v = pos  
        rs.setvalue( "col1", v )  
        v = pos * 11  
        rs.setvalue( "col2", v )  
        v = pos * 111  
        rs.setvalue( "col3", v )  
  
        rs.update()  
  
        pos = pos + 1  
    Wend  
End Sub
```

Example ex_db_insert.bas: Insert a new row into table tab1

Delete current database row

Position the edit-cursor on the db record that should be deleted and call the method *delete* for the recordset component.

```
t.Delete()
```

Delete the current row from table tab1. Since we do not position the row-pointer we delete the first row in the table.

```
Dim v As Variant  
  
Sub button1_clicked()  
    Dim dbc As dbconnection  
    Dim rs As recordset  
  
    dbc.connect( "hbasic", "root", "" )  
    rs.open( dbc, "tab1" )  
  
    rs.delete()  
End Sub
```

Example ex_db_delete.bas: Delete the first recordset from table tab1