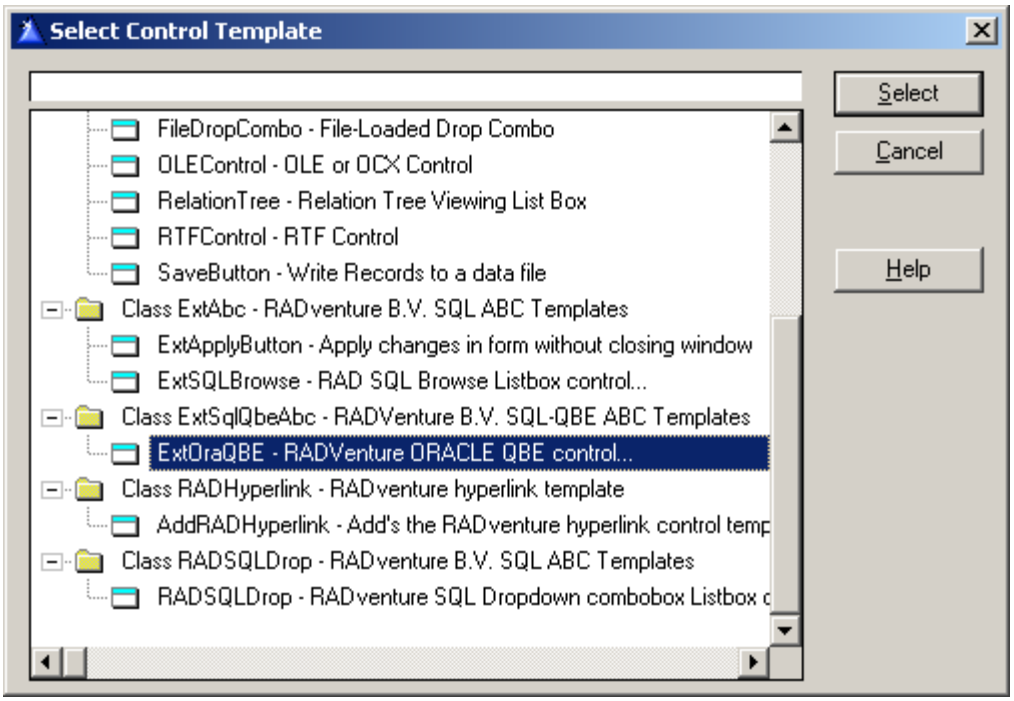
RADventure SQL QBE Control Template
Programmers Documentation

Table of contents

1 The RAD SQL QBE Control Template

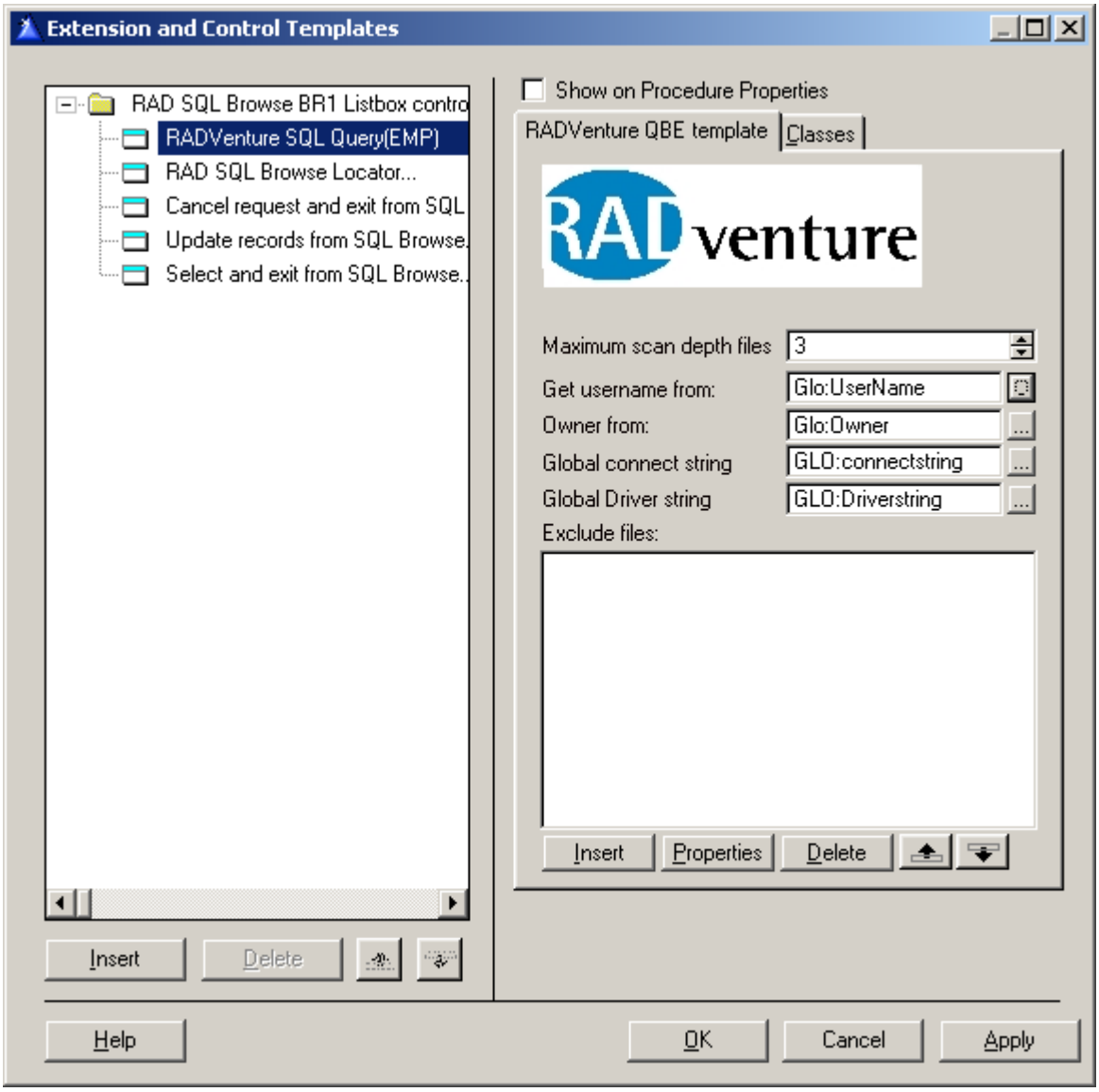
1)
Open a procedure that contains a RAD SQL Listbox Control (REQUIRED!).
a) Open the Window Formatter
b) Click on the menu ‘Populate’ then ‘Control Template’

2)
The following screen representation is a sample of the screen that appears at this point (depending on your own loaded control templates)



3)
After you have selected the SQL QBE control you place three buttons on the window : a QBE-button, a QuickQBE-button and a Reset-button. If you only want to use one of the QBE options (QBE or QuickQBE) you can remove the other button without removing the whole control template.

When you, after right mouse clicking, select ‘Actions’, or after leaving the Window Formatter and click the ‘Extensions’ button you will see the programmers input needed and the options for these controls:



Maximum scan depth files
Here you can enter to what depth in the file relations of the primary file the related tables should be added for use in the query.
Get username from
Select the global (or local) variable which contains the username.
If you use the RADventure ConnectionClass you can leave this field empty. The ConnectionClass.GetUserName method will then be used.
Owner from

Select the global (or local) variable which contains the owner name. This can be the same as the username.

If you use the RADventure ConnectionClass you can leave this field empty. The ConnectionClass.GetOwner method will then be used.

Global connect string

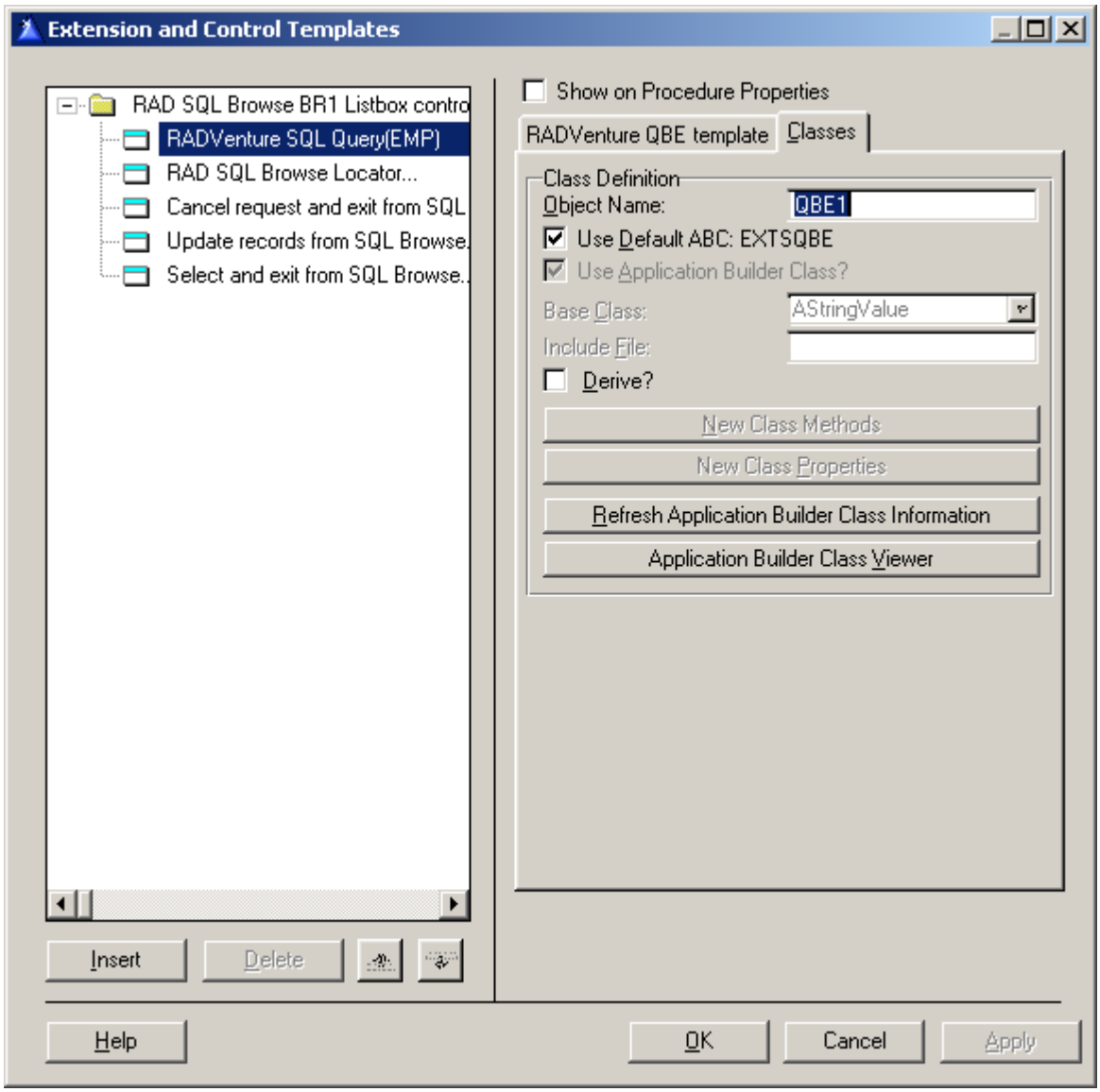
Select the global variable which contains the connect string, usually "UserName/Password@Server".

Global driver string

Select the global variable which contains the driverstring. This variable can be omitted.

Exclude files

If you want to exclude files from the query you can here insert these files.

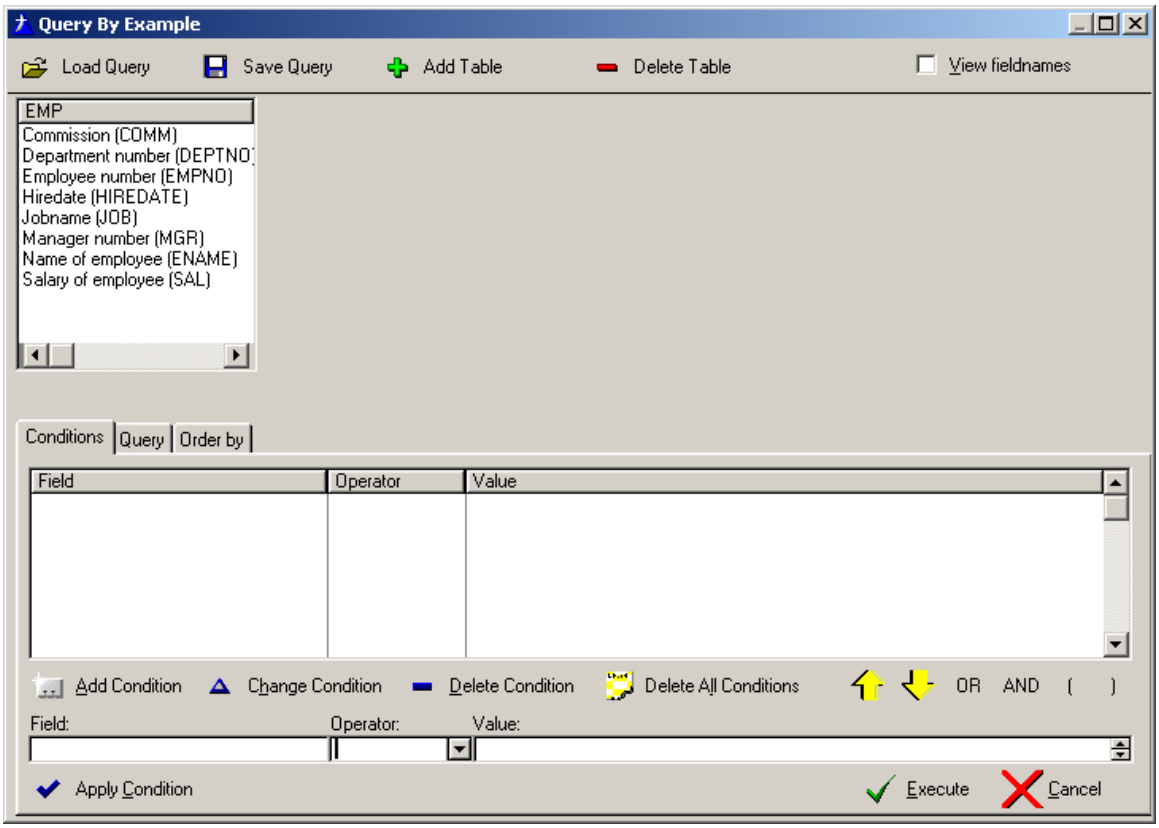


The RAD SQL QBE control is based on the EXTSQBE Class, using the RADSQBE.INC and RADSQBE.CLW source files.

When you have added the SQL QBE control (QBE and/or QBEQuick and Reset buttons) to your SQL Browse and you have entered the programmer's input (see previous screens) you are ready to see the SQL Query By Example at work.

2 The RAD SQL QBE At Work

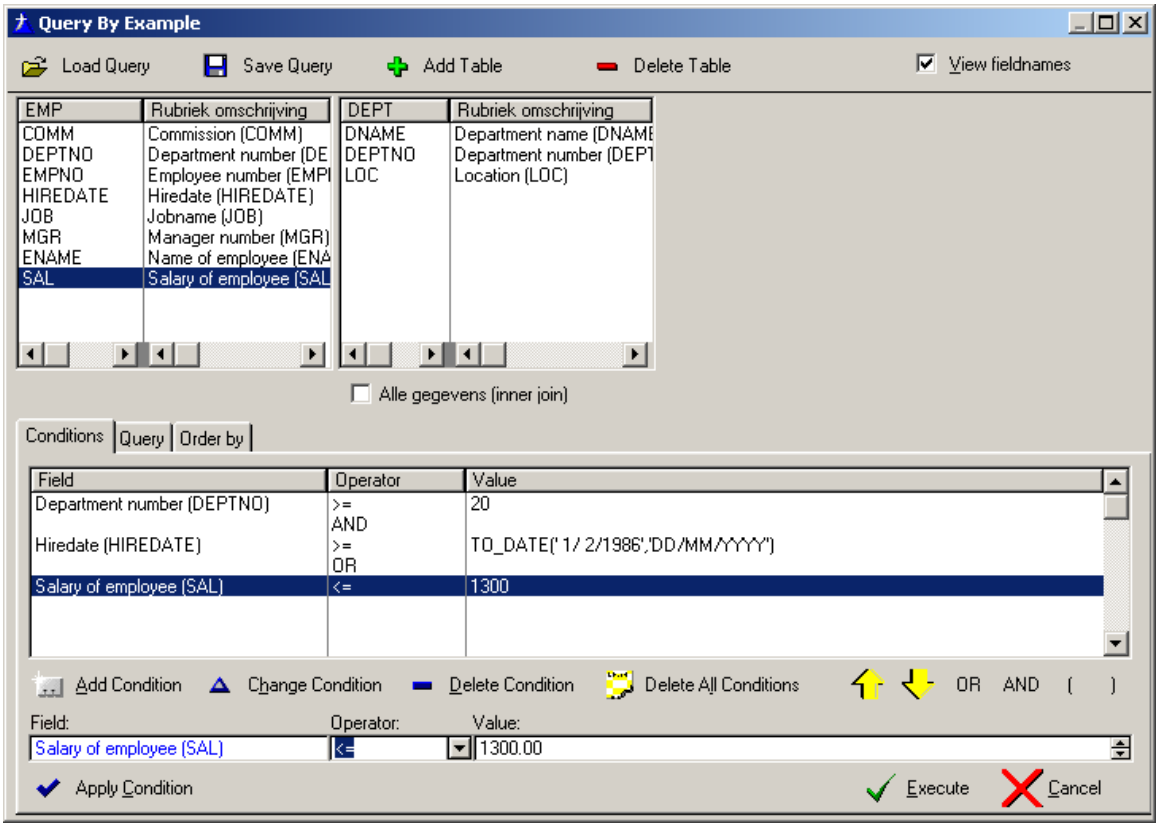
Run your application and go to the SQL Browse with the QBE control.
When you press the "QBE" button you will see this window:



By double clicking on the table fields (in the upper left listbox) you can start making a condition for the query.
You can add a (related) table (dependant on the file relations and the maximum scan depth you entered on the actions tab of the QBE control) : a new listbox with the fields of that table will appear and you can use these fields for your query.
Subsequently when you delete a table the listbox with the fields of that table will disappear from this window.
When you have made (the conditions of) a query you can save this query by pressing the "Save query" button. The query will be added to a table called "SavedQueries" on your SQL server.

By pressing the “Load Query” you can select and load a previous saved query for executing on your SQL Browse.
Default the field description will be shown in the field’s listbox. These descriptions are read from the comments in the “All_Col_Comments” table. When you also want to see the fieldnames check the “View fieldnames” checkbox on the upper right of this window. (If there are no comments in the “All_Col_Comments” table for a field the fieldname will be shown and checking the “View fieldnames” will result in showing the fieldname twice)

The next screenshot shows the same window after adding a table (an extra field listbox), checking the “View fieldnames” checkbox and adding some conditions to make a query statement.



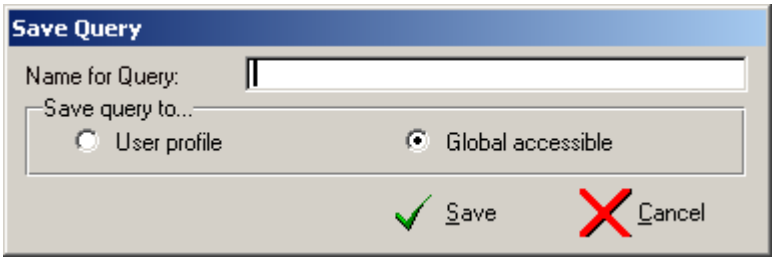
On the lower part of the window there are tree Tabs:
The “Conditions” tab shows the conditions that together make the query statement.
The “Query” tab shows the SQL query statement based on these conditions.

The “Order by” tab gives you the option to add an ORDER BY condition. Default it takes the current order from the SQL browse. You can omit ORDER BY by unchecking the “Use order by” checkbox or change the ORDER BY by entering a different statement (<Table.Field1,Table.Field2,..>).

After you completed the query statement (the conditions) you press the “Execute” button to return to the SQL Browse where your query statement will be used to reset the browse and show you the result set of records.
If you press the “Cancel” button the query statement will be cleared and you return to the SQL browse.
If you press the “Reset” button on the SQL browse window the query statement will be cleared and the browse will be reset.

3 Saving a query

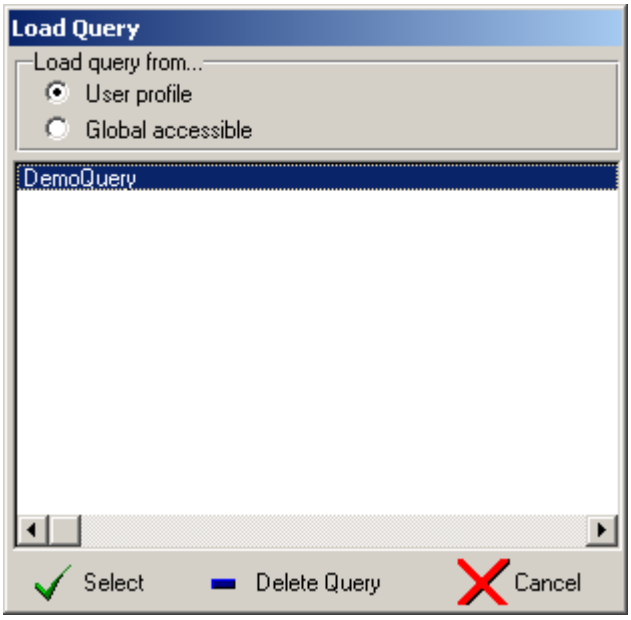
From the QBE window you can save a query (set of conditions) by pressing the “Save query” button. The next window will appear:



You must enter a name for your query and select one of the two options of “Save query to”. These options are:
User profile : the query will be saved under the username and will only be accessible for this user when you want to load the saved query.
Global accessible : the query will be saved with ‘*SYS’ as username. This makes it possible that other users can load this query when they press the QBE button from the same SQL browse procedure.
If the queryname already exists you will be asked if you want to overwrite the old query. If you press ‘Yes’ the old query will be deleted and the new one saved. If you press ‘No’ you will return to the “Save Query” window to change the name.

4 Loading a query

From the QBE window you can load a previous saved query by pressing the “Load query” button. The next window will appear:

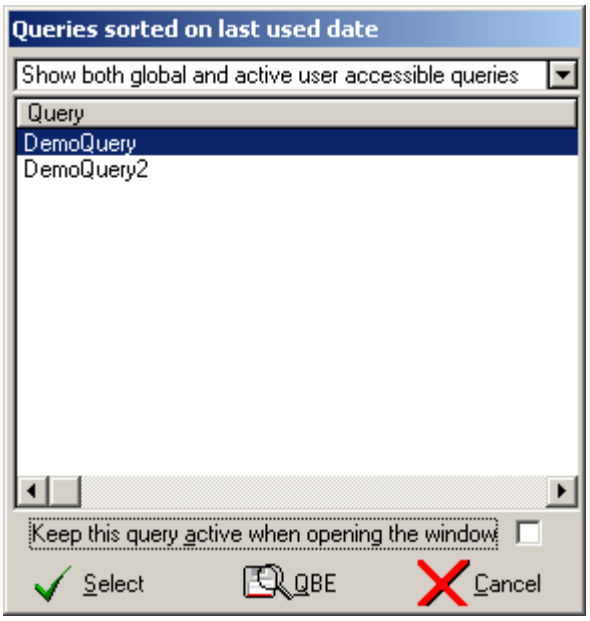


When you select “User profile” you will see all queries that where saved under the username. When you select “Global accessible” all queries the different users saved with the option “Global accessible” will be shown.
When you press “Select” or doubleclick in the listbox on the queryname that you want to select you return the QBE window and the conditions of the loaded query are visible in the conditions listbox. You can immediately execute this loaded query by pressing the “Execute” button or first make some changes to the loaded conditions.
When you press “Delete query” the selected query will be deleted from the SavedQueries table.

5 The RAD SQL Quick QBE At Work

When you use the SQL QuickQBE button there are some differences with the “normal” QBE control.

Run your application and go to the SQL Browse with the Quick QBE button.
When you press the “QuickQBE” button you will see this window:



The Quick QBE goes immediately to a selecting window with previously saved queries, so you can load a saved query directly without first going to the QBE window.

You can choose from three options in the upper combo box:
“Show all global accessible queries” : all queries saved for the SQL browse procedure with the option ‘Global accessible’ will be shown.
“Show only queries accessible to active user” : only those queries that where saved under the active username will be shown.
“Show both global and active user accessible queries” : both kind of queries will be shown.

By pressing the “Select” button or double-clicking on the query you want to select the query statement will be generated and you will return to the SQL browse. The query

statement will be executed and the browse will be reset to show the result set of records that are valid for the query statement.

If you press the “QBE” button you go to the QBE Window, where you can make a query statement. (See the RAD SQL QBE At Work above).

By pressing the “Cancel” button you will return to the SQL browse window without a query statement.

If you want a query statement to be executed immediately at the opening of the SQL browse window you have to select that query and check the option “Keep this query active when opening the window”. Each time the SQL browse window is opened it looks in the SavedQueries table for a query with the Keep_Active attribute. If found the browse will be opened with the query statement of that query.

6 SQL QBE Manager

7 SQL QBE Managers Methods

Init (Initialize QBE object)

Init (In_SQLBrowse, In_Owner, In_Primary, In_Connectstring, <In_Driverstring>)

In_SQLBrowse
Class
Reference to the (RAD SQL) browse object
In_Owner
String
Owner name (used in theGetTableFields method)
In_Primary
String
Primary file in the browse
In_ConnectString
*?
Reference to the connectstring (used for loading and saving queries)
In_DriverString
*?
Reference to the driverstring

The Init method is called from the procedure where the QBE control template is added. The QBE object will be initialized and the references to the (Field Equate Labels of the) QBE- , QuickQBE- and Reset buttons will be made.

AddQueryTable (Add table to list of tables that can be used to create QBE statement)

AddQueryTable (In_Tablename)

In_TableName
String
Tablename

The AddQueryTable method is called from the procedure where the QBE control template is added. Depending on the file relations of the primary file in the browse and the depth to go into these relations a number of tables can be added. These tables can be used to make the QBE statement in the QBE window.

GetPrimaryFields (Get fields of the primary file of the SQL Browse)

GetPrimaryFields ()

The GetPrimaryFields method is called from the QBE method, when opening the QBE window for the first time. The fieldnames and descriptions of the primary file are fetched by calling the GetTableField method for the primary file. These fields will then be shown in a listbox when the QBE window opens.

GetTableFields (Get fields and field descriptions of a table)

GetTableFields (In_Tablename)

In_Tablename
String
First tablename

The GetTableFields method is called from several other methods. The fieldnames of the table are fetched from the "ALL_ TAB_COLUMNS" table and the field descriptions from the "ALL_COL_COMMENTS" table. These fields and field descriptions will then be shown in a listbox in the QBE window.

SelectTable (Let user select table needed for creating QBE statement)

SelectTable (), Byte

The SelectTable method is called from the QBE method, when the user pushes the "Add Table" button. A selecting window with the QBE tables will be opened. If the user selects a table the GetTableFields method is called and the returnvalue is set to TRUE. In the

QBE window a new listbox will be created with the fields of the added table. If the user cancels the returnvalue is FALSE and no listbox will be added.

DeSelectTable (Let user deselect table)

DeSelectTable (), Byte

The DeSelectTable method is called from the QBE method, when the user pushes the "Delete Table" button. A selecting window with the previously added tables will be opened. If the user selects a table the table is removed from the added table list and the returnvalue is set to TRUE. The listbox will be removed from the QBE window. If the user cancels the returnvalue is FALSE and the listbox will not be removed.

QBE (Call QBE window for creating the QBE statement)

QBE (), Byte

*The QBE method is called from the procedure where the QBE control template is added, when the user pushes the "QBE" button. The QBE window will be opened. The user can create a QBE statement, load a previously saved query and more (See **The QBE At Work**). When the user pushes the "Execute" button the QBE statement is generated and the returnvalue is TRUE. In the SQL browse the QBE statement will be executed. If the user pushes the "Cancel" button the QBE statement is cleared and the returnvalue is FALSE.*

SaveQry (Save the query / QBE statement)

SaveQry ()

The SaveQry method is called from the QBE method, when the user pushes the "Save Query" button. A window will be opened and the user can enter a name and choose between saving the query under the username or global accessible. If the "Save" button is pushed the query will be added in the "SavedQueries" table.

LoadQry (Load a previously saved query / QBE statement)

LoadQry (), Byte

The LoadQry method is called from the QBE method, when the user pushes the "Load Query" button. A window will be opened with the previously saved queries fetched from the "SavedQueries" table. If the "Select" button is pushed the returnvalue is TRUE and the query will be shown in the conditions listbox in the QBE window. If the user cancels the returnvalue is FALSE.

Destruct (Destruct QBE object)

Destruct ()

The Destruct method disposes of the queues used by the QBE object .

CheckSqlFormula (Check if created QBE statement is valid)

CheckSqlFormula (), Byte

The CheckSqlFormula method is called from the QBE method, when the user pushes the "Execute" button. The created QBE statement (the conditions) is checked and if there is some error a message will be shown and the returnvalue is FALSE. The QBE window will not be closed and the QBE statement will not be executed. If no error is found the returnvalue is TRUE and the QBE window will be closed and the QBE statement will be executed.

QuickQBE (Call selection window with previously saved queries)

QuickQBE (), Byte

The QuickQBE method is called from the procedure where the QBE control template is added, when the user pushes the "QuickQBE" button. A selecting window with previously saved queries will be opened. If the user selects a query the QBE statement is generated and returnvalue is TRUE. The window is closed and the QBE statement will be executed. If the user pushes the "Cancel" button the returnvalue is FALSE.
If the "QBE" button is pushed the QBE method is called (opening QBE window). If the QBE method returns TRUE ("Executed" is pushed) the QuickQbe window is closed and the QBE statement will be executed, else the window stays open.
The user has also the option to set the Keep_Active attribute for a query. This means that the query will be executed when opening the SQL browse (see CheckForActiveQueries)

MakeDoubleQuotes (Make quotes for a string value in the QBE statement)

MakeDoubleQuotes (Waarde, NumberOfQuotes=2), String

Waarde
String
The value of the string
NumberOfQuotes
Byte
Number of quotes, default 2

The MakeDoubleQuotes method is called from the QBE method, when formatting a (string) value of a condition. If the string value contains quotes it returns the value with double quotes.

CheckForActiveQueries (Check if a saved query has the Keep_Active attribute)

CheckForActiveQueries ()

The CheckForActiveQueries method is called from the procedure where the QBE control template is added, at the window initialization. If there is a query (in the "SavedQueries" table) for the active procedure and username with the Keep_Active attribute the QBE statement will be generated and executed at opening the SQL browse .

8 RADventure Support

RADventure Support	
Email:	tools@radventure.nl
Telephone:	+31 (0)346 29 09 80
Fax:	+31 (0)346 29 09 08
Post:	PO Box 1069, 3600 BB Maarssen, The Netherlands

9 Copyright