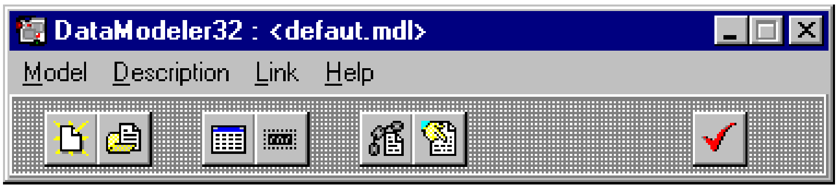
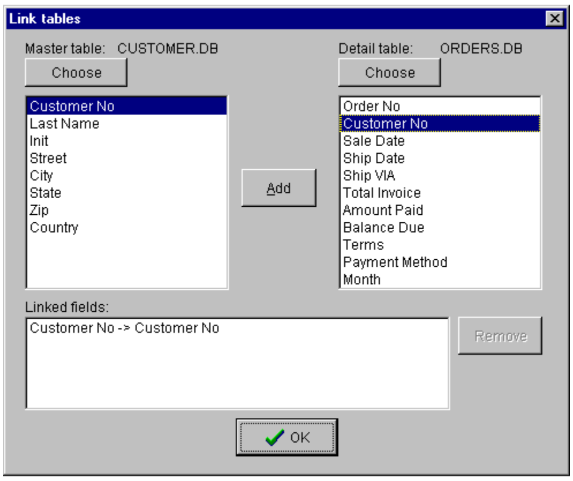
Review: MK QueryBuilder Version 1.22

Reviewed by Tim Boyd



➤ *Above: Figure 1*

➤ *Right: Figure 2*



Link tables

Master table: CUSTOMER.DB

Choose

Customer No
Last Name
Init
Street
City
State
Zip
Country

Add

Detail table: ORDERS.DB

Choose

Order No
Customer No
Sale Date
Ship Date
Ship VIA
Total Invoice
Amount Paid
Balance Due
Terms
Payment Method
Month

Linked fields:

Customer No -> Customer No

Remove

✓ OK

MK QueryBuilder (MKQbe) is a component that makes it very easy to add Query By Example (QBE) capabilities to Delphi applications (and others). This makes it possible for users to run pre-programmed queries, create their own queries, review the results on screen or in reports and export the retrieved records into a variety of formats. It is a product of the MK Organisation in France.

Features

MKQbe comes with both 16-bit and 32-bit versions, on two diskettes. For this review, the 32-bit version was tested with Delphi 2.0. The installation process is straightforward for both the component and help files. The setup program also provides for the installation of optional demo files that are useful for quickly exploring MKQbe's features. Its README file cautions anyone installing the 16-bit version to increase the default compiler stack size by 5Kb to 10Kb. For anyone developing multi-lingual systems, the program can be installed in French, as one might expect considering the vendor's country of origin. A typical Delphi installation uses less than 1Mb of disk space.

As a side note, the program's French ancestry sometimes shows itself in interesting ways. The *Creating a new data model* section of the developer's manual, for example, shows a sample dialog box in French. Due to its compliance with standard dialog box conventions, however, even I can figure out that `Nom de fichier` must mean `File name` and `Dossiers` is used to select folders. This is simply a documentation error – the actual dialog box is in English.

The program includes two modules:

- DataModeler: used by the developer to create pre-defined relational data models which are saved in a model (MDL) file for input into the query module.
- QBE: the query module that is integrated into the final Delphi application.

The first step in adding query capability to an application is to run the DataModeler module, which is a stand-alone executable (Figure 1). You then add tables to the data model, selecting them either by alias or by directory. After selecting a table you have the option to assign a more user-friendly name to it.

The next step is to define the fields that are to be available for query purposes. Again, user-friendly alternative field names can be assigned.

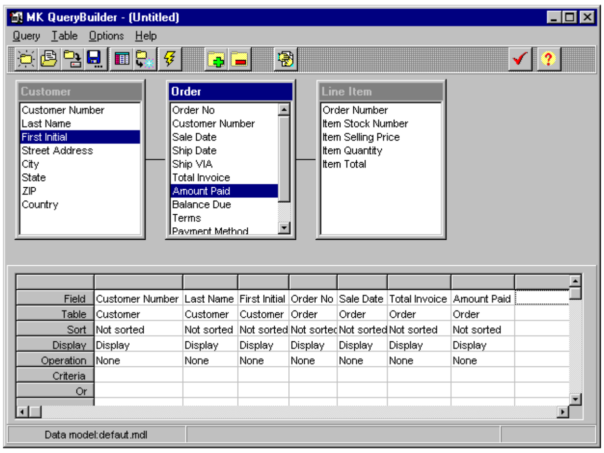
Next, you define links between the tables (as shown in Figure 2). The developer's manual warns you at this point that *The system does not check the consistency of these links. It is up to you to make sure the various field types are compatible, and that the link makes logical sense..*

Having defined the tables, fields and links, the next step is to place a (non-visual) MKQbe component into a Delphi application and assign appropriate property values, which is really easy to do. Running the query in your application is a simple one-liner: `MKQbe1.Run`.

The runtime MK QueryBuilder window is shown in Figure 3 (next page). I selected `Table|Add` from the menu, then chose the `Customer`, `Order` and `Line Items` tables. This resulted in each of the three tables being displayed in separate windows within the form A single horizontal line symbolizes the links between them. To select fields for the query, you just double click the desired fields within each table window.

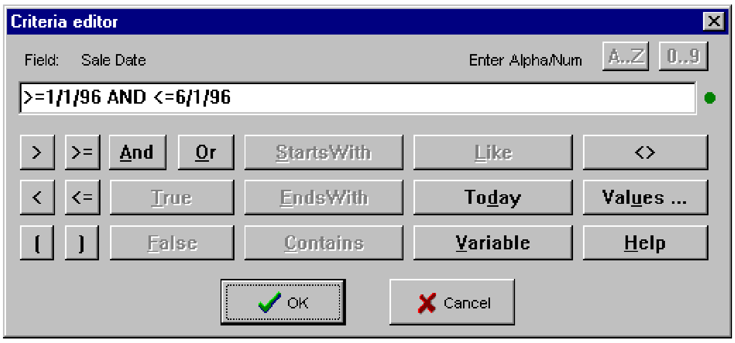
The resulting query columns can easily be dragged to a different position or deleted. To set query parameters, simply right-click or double-click the appropriate `Criteria` cell to access the Criteria Editor through a pop-up menu (see Figure 4). While I found it to be frustrating testing queries in debug mode (an *invalid date* warning was triggered for each keystroke of the date values), the editor works without a hitch when running in a stand-alone executable.

As you can see in Figure 4, the editor provides handy buttons for



Field	Customer Number	Last Name	First Initial	Order No	Sale Date	Total Invoice	Amount Paid
Table	Customer	Customer	Customer	Order	Order	Order	Order
Sort	Not sorted	Not sorted	Not sorted	Not sortec	Not sorted	Not sorted	Not sorted
Display	Display	Display	Display	Display	Display	Display	Display
Operation	None	None	None	None	None	None	None
Criteria							
Or							

Data model:defaut.mdl



Field: Sale Date

>=1/1/96 AND <=6/1/96

➤ *Left: Figure 3*

➤ *Above: Figure 4*

➤ *Below: Figure 5: The Client Reaction Rating!*

Rating	Client Reaction
1	Where is that programmer? Track him down and kill him!
2	Why was this included? Oh well, guess it won't hurt anything.
3	It does serve a basic purpose. Some of us actually use it.
4	Nice job – most of us really like it!
5	Where is that programmer? Track him down and knight him!

specifying typical query criteria. When the user has specified their selection criteria, the query is executed by clicking the lightning bolt button. The query can be saved for future re-use.

It also is possible to define selection criteria that cause a dialog box to prompt the user for appropriate values when the query is executed. This makes it possible to insulate unsophisticated users from unwanted details by pre-defining queries that prompt for all input criteria in a user-friendly manner – a nice feature.

The MK QueryBuilder window also has a help facility available to assist users in creating their own queries. Like most help facilities, however, it could benefit from more illustrated examples.

The `Options` menu allows users to select only unique values or to show the total record count of the answer file. Calculating the record count can increase query run time by a factor of 10 or more, so the README file contains instructions for selecting an option to handle this situation on your server. Unfortunately, the default option is to calculate the record count, a feature that should definitely be changed in the next release.

Query results are displayed in a read-only grid that is not refreshed until the query is re-run. A toggle button can be used to display records one at a time in a dynamically generated form that has a nice 3D effect. The query result screen provides convenient features for navigation, including a search facility. It also provides options to save the query results in Paradox, dBase, ASCII or Excel formats. You can even copy the results to the clipboard.

Query results can be printed in report form. The print setup dialog even allows you to select the fields that are to appear in the report. Report formatting features such as header/footer, page numbering and borders are provided. Strangely enough, however, there is no provision for placing a date and/or time string in the report header or footer. Also, I found the margins as printed don't match the requested setting and the title fonts feature simply doesn't work. The report facility definitely has room for improvement in future releases.

Distributing an MKQbe application is somewhat complex, requiring a modification to the user's WIN.INI and the installation of your application files, four DLL files, two VBX files, an HLP file and a special LIC license file that enables the application to work with the pre-defined data models.

When distributing applications containing MKQbe, I suggest that users should be cautioned to limit the scope of their queries and to be aware of the performance impact of showing the total record count in the query result. Otherwise, they may be staring at a busy hourglass cursor for a long time.

Conclusion

If you are thinking of providing Query By Example capabilities in your programs, MK QueryBuilder is worth considering. For any tool that allows users to create and run their own queries, however, it is wise to include documentation and/or training that encourages the use of indexed fields for selection criteria, the avoidance of certain operators (such as `NOT`) and exercising caution when considering multiple table joins. In many cases, it may be wise to provide only pre-programmed queries with easy-to-use dialog boxes that prompt the user for the desired selection criteria.

The most important factor I consider in selecting components for inclusion in Delphi applications is the anticipated reaction of my clients. Therefore, I use a "Client Reaction" rating scale that reflects my own feverish imaginings as to how the new features will be received (Figure 5). Based on this scale, I rate MK QueryBuilder as a 3.5 with strong potential to move higher in future versions.

MK Query Builder is available from software tool retailers. In the USA, street price is $179. Tim Boyd is from Simi Valley in California, email: timboyd@warpweb.co