

# **F r a m e S c r i p t**

## **R e t r i e v a l**

### **D a t a b a s e O b j e c t**

### **R e f e r e n c e**

**ELMSOFT INC.**  
7954 Helmart Drive  
Laurel Maryland 20723  
USA

FrameScript Database Retrieval Manual version 2.00  
November 2000

Copyright © 1997-2000 ElmSoft, Inc. All rights reserved.

ElmSoft, Inc. ("ElmSoft") and its licensors retain all ownership rights to the FrameScript computer program and other computer programs offered by ElmSoft (hereinafter collectively called "ElmSoft Software") and their documentation. Use of ElmSoft Software is governed by the license agreement accompanying your original media. The ElmSoft Software source code is a confidential trade secret of ElmSoft. You may not attempt to decipher, decompile, develop, or otherwise reverse engineer ElmSoft Software, or knowingly allow others to do so. Information necessary to achieve the interoperability of the ElmSoft Software with other programs may be available from ElmSoft upon request. You may not develop passwords or codes or otherwise bypass the security features of ElmSoft Software.

This manual, as well as the software described in it, is furnished under license and may only be used or copied in accordance with the terms of such license. The information in this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by ElmSoft. ElmSoft assumes no responsibility or liability for any errors or inaccuracies that may appear in this book.

Except as permitted by such license, no part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of ElmSoft.

Please remember that existing artwork or images that you may desire to scan as a template for your new image may be protected under copyright law. The unauthorized incorporation of such artwork or images into your new work could be a violation of the rights of the author. Please be sure to obtain any permission required from such authors.

FrameScript and ElmSoft are trademarks of ElmSoft.

Adobe, the Adobe logo, Acrobat, Acrobat Exchange, Adobe Type Manager, ATM, Display PostScript, Distiller, Exchange, Frame, FrameMaker, FrameMaker+SGML, FrameMath, FrameReader, FrameViewer, FrameViewer Retrieval Tools, Guided Editing, InstantView, PostScript, and SuperATM are trademarks of Adobe.

IN NO EVENT WILL APPLE, ITS DIRECTORS, OFFICERS, EMPLOYEES, OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE APPLE SOFTWARE EVEN IF APPLE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

The following are copyrights of their respective companies or organizations:

PANTONE® Computer Video simulations displayed may not match PANTONE-identified solid color standards. Use current PANTONE Color Reference Manuals for accurate color.

Six-color Process System Patent Pending - Pantone, Inc.

PANTONE Open Color Environment™ (POCE™) © Pantone, Inc., 1994, 1996.

The following are trademarks or registered trademarks of their respective companies or organizations:

Apple, AppleLink, AppleScript, AppleTalk, Balloon Help, Finder, ImageWriter, LaserWriter, PowerBook, QuickDraw, QuickTime, TrueType, XTND System and Filters, Macintosh, and Power Macintosh are used under license / Apple Computer, Inc.

Microsoft, MS-DOS, Windows / Microsoft Corporation

Sun Microsystems, Sun Workstation, TOPS, NeWS, NeWSprint, OpenWindows, SunView, SunOS, NFS, Sun-3, Sun-4, Sun386i, SPARC, SPARCstation / Sun Microsystems, Inc.

All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Written and designed at Elmsoft, Inc., 7954 Helmart Drive, Laurel, MD 20723, USA

For civilian agencies: Restricted Rights Legend. Use, reproduction, or disclosure is subject to restrictions set forth in subparagraphs (a) through (d) of the commercial Computer Software Restricted Rights clause at 52.227-19 and the limitations set forth in ElmSoft's standard commercial agreements for this software. Unpublished rights reserved under the copyright laws of the United States. The contractor/manufacturer is Elmsoft, Inc., 7954 Helmart Drive, Laurel, MD 20723, USA.

# Database Reference Contents

- 1 Database Reference Contents - - - - - 3**
  
- 2 Overview - - - - - 5**
  - Introduction - - - - - 5
  - EDB Object - - - - - 5
  - EQUERY Object - - - - - 5
  - ODBC Setup - - - - - 5
  - Demo Database - - - - - 9
  - Demo database scripts - - - - - 9
  
- 3 Database Objects - - - - - 11**
  - EDB Object - - - - - 11
    - Method Descriptions - - - - - 12
    - New - - - - - 12
    - Open- - - - - 12
    - Close - - - - - 13
    - Delete - - - - - 13
  - EQuery Object - - - - - 13
    - Method Descriptions - - - - - 14
    - New - - - - - 14
    - RunQuery - - - - - 15
    - RunSqlTables- - - - - 15
    - GetNext - - - - - 15
    - Delete - - - - - 15



# Chapter 2

## Overview

### Introduction

This function allows you to retrieve data from standard ODBC compliant databases via a FrameScript script. You may use SQL commands to perform searches and get results set, from which you may access data variables the same way you would use standard FrameScript variables.

The data base retrieval objects (EDB and EQUERY) are implemented as external objects. These objects are located in one external file called eplus.dll. To use these objects, the file must be registered to FrameScript. See the Options dialog box in the FrameScript Scriptwriter's guide for information on registering external objects.

### EDB Object

The EDB object represents an entire database. You use this object to identify, open and close the database. You supply the object with a data source name (defined when you register your database to the ODBC manager (see below), and optionally a User ID and Password. Then you connect to the database using the Open method (a subroutine call). The Close method terminates the connection.

### EQUERY Object

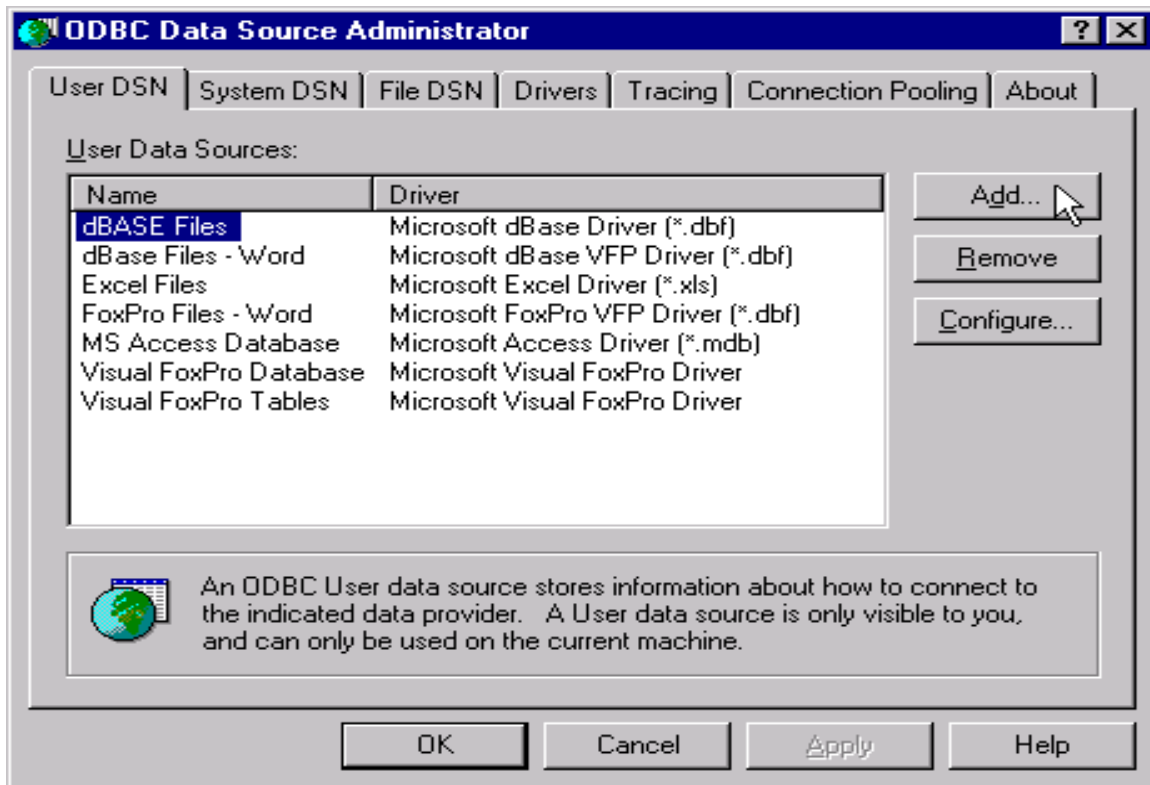
The EQUERY object represents a query and search result. Each EQUERY object is associated with an EDB object. After creating an EQUERY object, you set its Select property to the SQL search that you wish, then you run the RunQuery method to perform the search. You access the Fields property to get the data from the current record and you may use the GetNext method to move forward to the next record in the results set.

### ODBC Setup

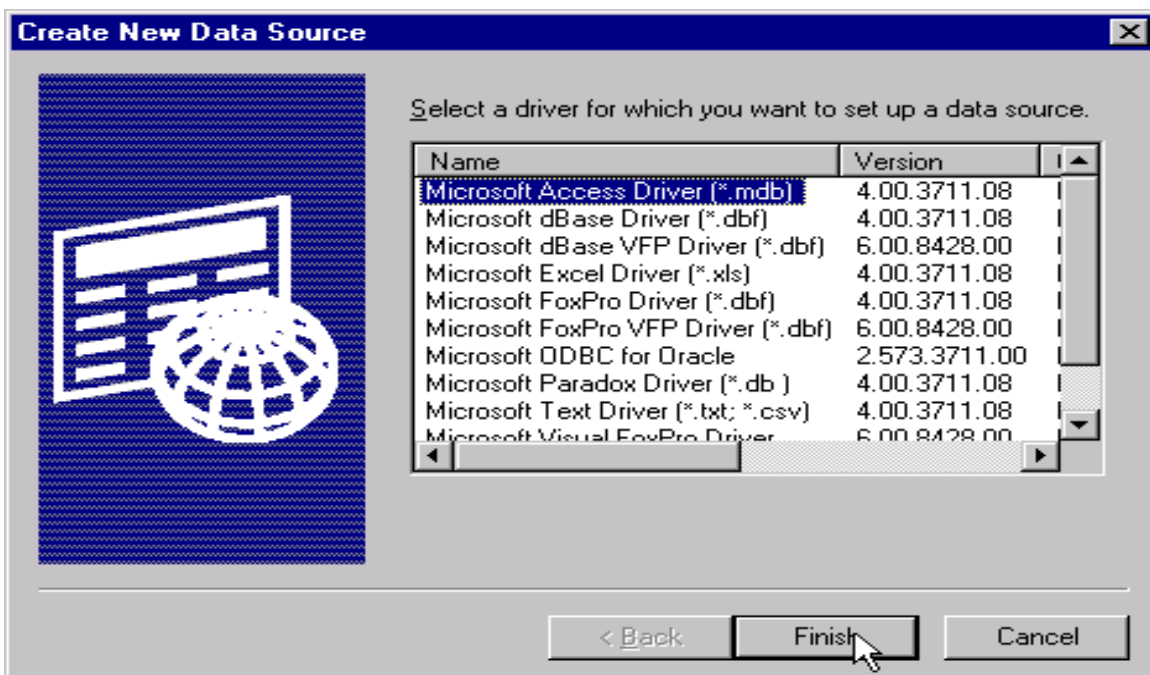
To use a database via the FrameScript objects you must first register that database with the ODBC Manager. This creates a Data Source. Access to ODBC databases is through data sources. To register a data source for the Demo Access 97 database (Issues.mdb) do the following:

## 1. Bring up the ODBC manager.

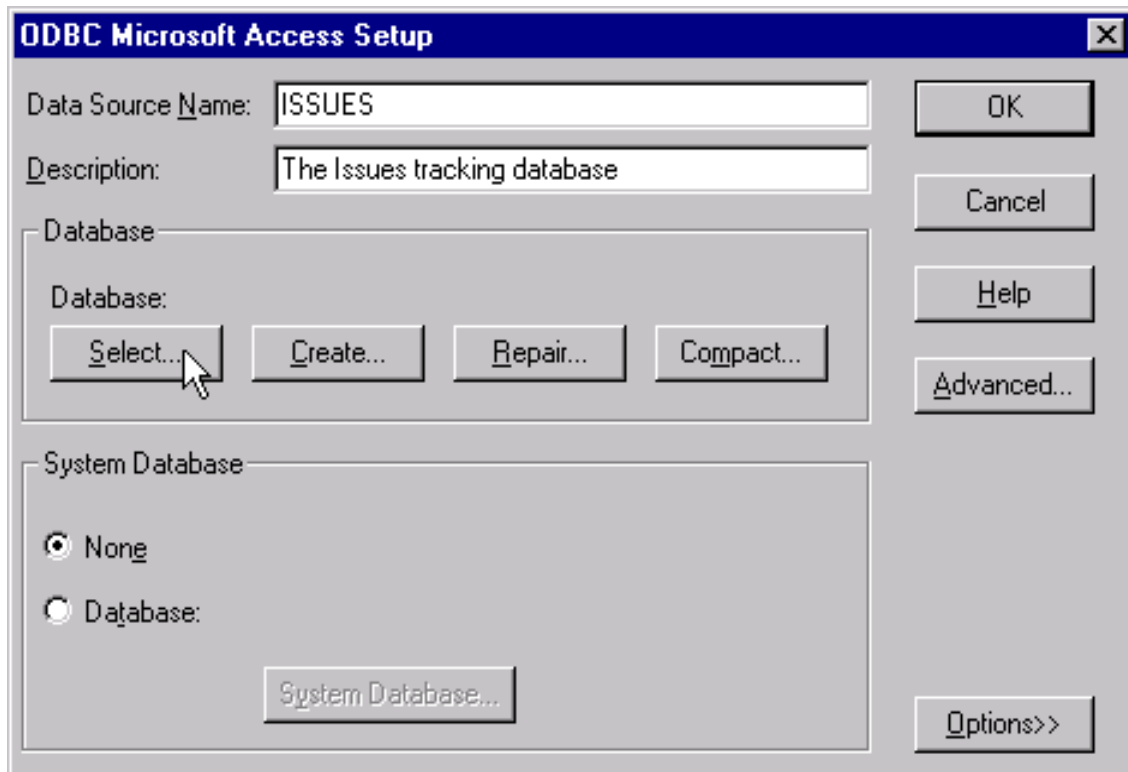
Click on the Start->Control Panel. When the Control panel comes up, double click on the ODBC icon and the following window (or one very much like it) should appear.



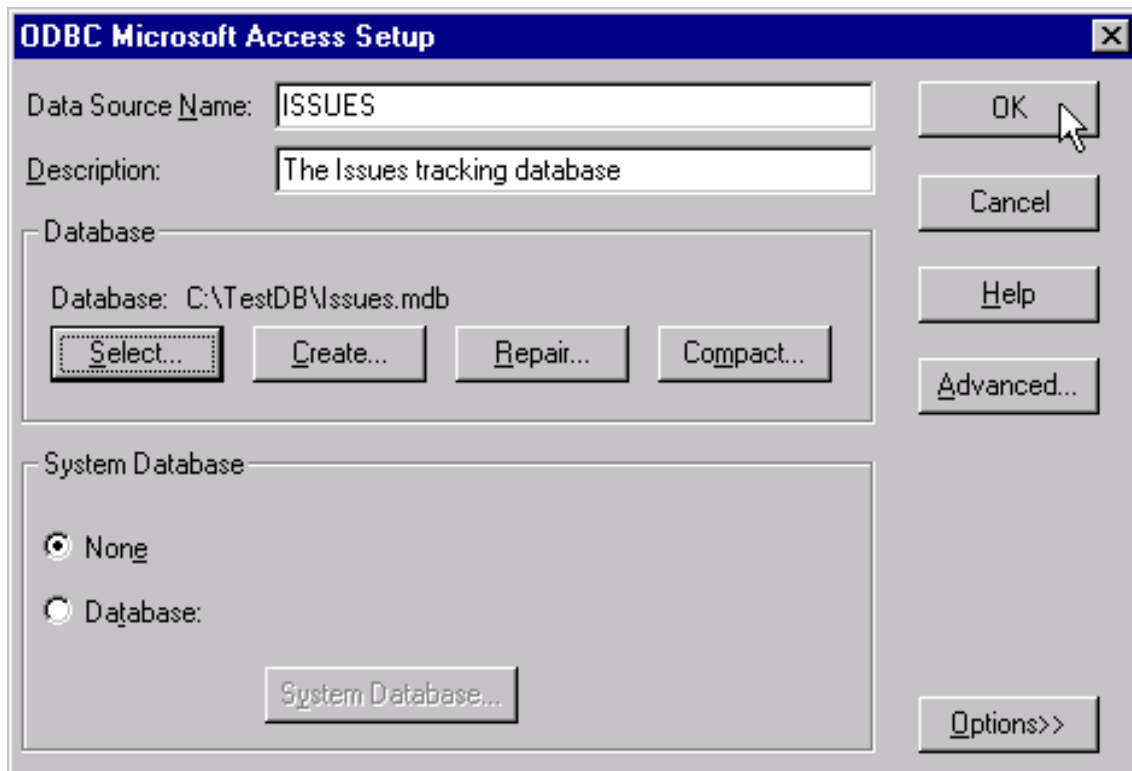
## 2. Click on the Add button to add a new data source. The following window should appear.



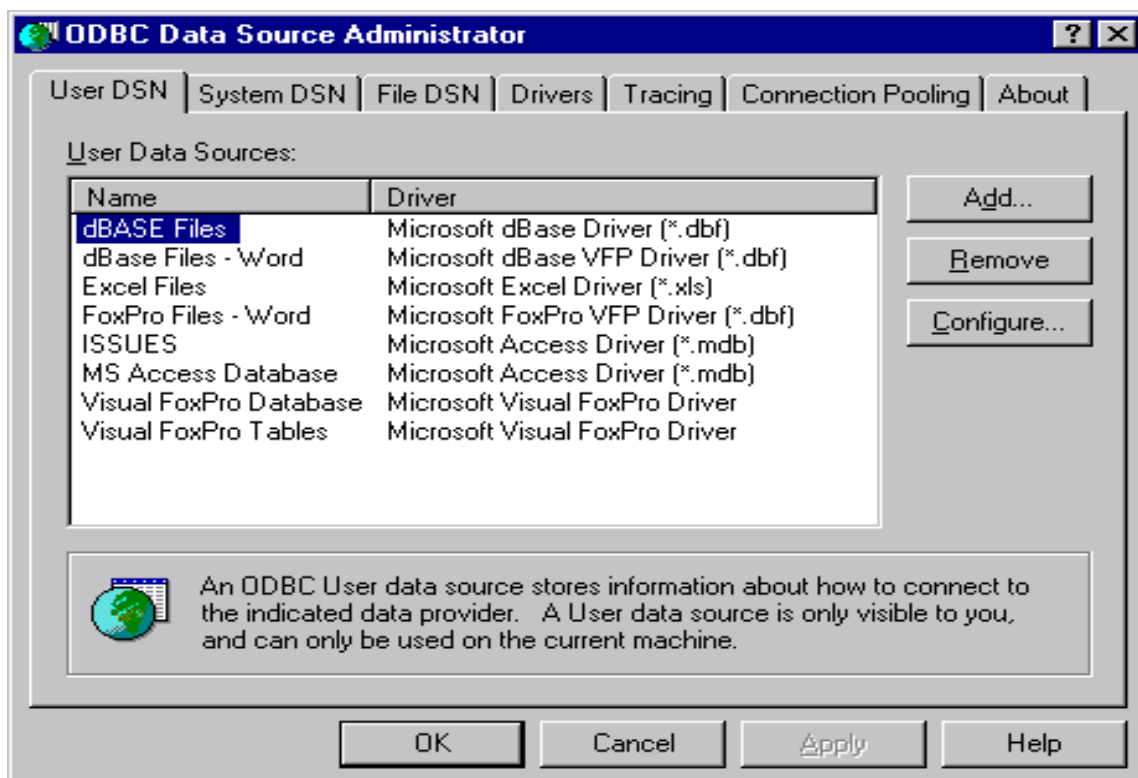
3. Select the driver for the database (in this case, it is Microsoft Access) and press Finish. The Microsoft Access Setup dialog will appear. Enter the name of the data source (in this case: ISSUES). This name is very important because it is used to identify the data source when you make a data connection in FrameScript. To run the demo database scripts make sure that you use the name ISSUES. The demo scripts expect a data source named ISSUES to exist. For your own databases, you can choose whatever name that you like. You may also optionally enter a description.



4. Press the Select button to choose the database file. The standard file selection dialog will appear. Go to the FrameScript directory and the DemoDatabase sub-directory and select the Issues.mdb file and press OK. The setup window should be updated as follows:



5. Press the OK to accept all the changes. The following window appears.

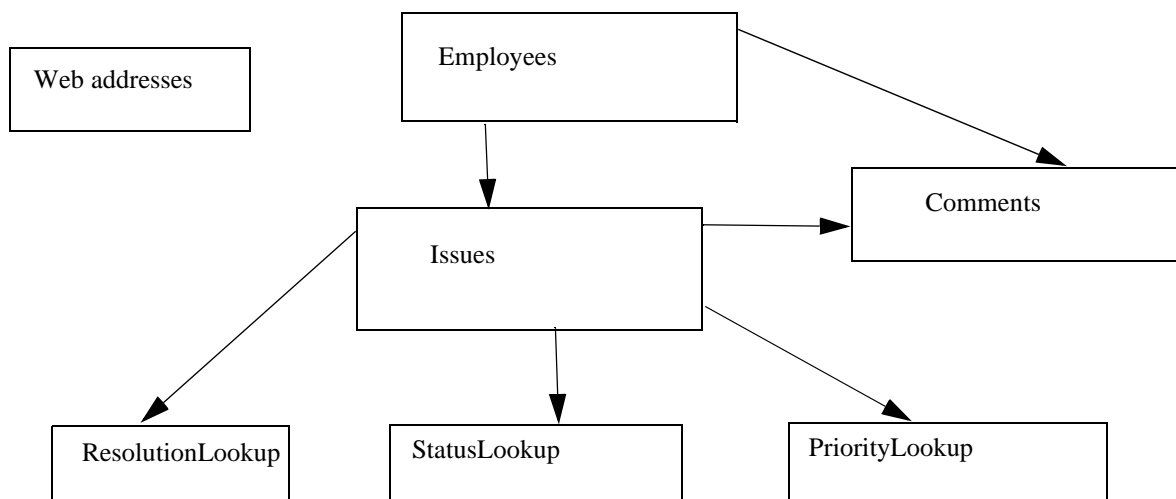


- The issues database now appears in the main window. Press OK to finish.

## Demo Database

The Demo Microsoft Access database (Issues.mdb) is a simple issues tracking database. Employees enter problems or issues and other employees make comments on the issue. The Issues table contains information about each problem identified. The employee table contains a list of employees. A comment table contains the employee comments about each issue. There are also smaller lookup tables, such as Resolution, Status and Priority.

The following diagram illustrates the relationship between the various tables.



## Demo database scripts

Now that the demo database has been registered to ODBC, you are ready to run some scripts that access that database.

The first script to run on a database is usually the general utility script called DBListFields.fsl. This script puts up a dialog box, which asks you to supply a Data source name and optionally a userid and password. When you click the OK button, the script builds a Frame document containing the names of all the tables for the data source and a list of fields for each table. It tells you what type of data the field contains, and, if you selected the data sample checkbox, it will include a line of data for each field (the first one in the table).

The following examples use the EDBUtils.fsl script that is found in the Lib directory. This is a set of database utilities.

### DBListFields.fsl

To run this script on the demo database which you have registered as a data source above, do the following:

- Start FrameMaker.
- Run the script (FrameScript->Run...) in the database Demo directory called DBListFields.fsl
- A dialog box should appear. Enter the name ISSUES in the data source field. Leave the UserId and Password fields blank. Check the Data sample checkbox.
- Press the OK button.

A FrameMaker document should be created containing a list of tables and fields.

### **DBIssuesReport.fsl**

The DBIssuesReport.fsl script is a simple report script. It illustrates how to open a database and read data from doing multiple searches. It generates a FrameMaker document which contains a line for each issue recorded in the database and for each issue it displays a list of comments for that issue.

To run this demo, do the following:

1. Start FrameMaker.
2. Run the script (FrameScript->Run...) in the database demo directory called DBIssuesReport.fsl.

A FrameMaker document should appear containing the report described above.

### **DBIssues.fsl**

The next demo script is much more complicated. It is an interactive script that fills in a table in the demo issues.fm document. The demo document (Issues.fm) is a form type document. It is designed to display the issues brought up by selected employee. You select an employee from a list of employees and the FrameScript script fills in the information in the table. This script also formats some of the data from the look up tables.

To run this demo, do the following:

1. Start FrameMaker.
2. **Install** the script (FrameScript->Install...) called Issues.fsl. When the dialog box appears asking for the name of the script, type in DBISSUES. Make sure that you install this script. Do not run it. It is an event script that stays around and responds to events.
3. Open the demo document located in the DatabaseDemo sub-directory called Issues.fm.
4. Click on the Select Employee button. A list of employees should appear. Select one and the screen will be updated with the new employee's information.
5. If you have Microsoft Access (97 or greater), you can open this database and make changes. Use the reload button to reload employee information.

# Chapter 3

## Database Objects

### EDB Object

The EDB object represents a database. You must create an EDB object using the NEW EDB command before attempting to open or otherwise use the database.

**Table 1: EDB Methods**

| Method Name | Method Description   |
|-------------|--|
| New         | Creates the EDB object.  |
| Open        | Opens the database to be represented by the object.                              |
| Close       | Closes the database to be represented by the object.                             |
| Delete      | Deletes the database object. Note that this does not delete the actual database. |

**Table 2: EDB Properties**

| Property Name | Data Type | Property Description  |
|---------------|-----------|---|
| IsOpen        | Integer   | True if a database is currently open; False otherwise.            |
| DataSource    | String    | The name of the datasource specified on the open method.          |
| ErrorCode     | Integer   | The FrameScript error code for the last method.                   |
| ErrorMsg      | String    | The Text of the last error.                                       |
| ODBCError     | Integer   | The ODBC error code for the last ODBC call.                       |
| User          | String    | The user name specified on the open method.                       |
| Password      | String    | The password specified on the open method.                        |
| DBMSName      | String    | The name of the database management system for the open database. |
| QuoteChar     | String    | The character used for quotes for this database.                  |

## Method Descriptions

### New

**Format:**

```
New EDB DataSource(datasrcname) [User(username)] [PassWord(password)]
NewVar(dbvar);
```

The New method creates a new EDB object. You must create an EDB object before attempting to open a database.

**Table 3: NEW EDB Options**

| Option Name | Option Description  |
|-------------|---|
| DataSource  | Identifies the ODBC data source name.   |
| User        | If the database has a User and Password, this option identifies the username. |
| PassWord    | If the database has a User and Password, this option identifies the password. |
| NewVar      | The name of the variable to hold the newly created object.                    |

**Example:**

The following script creates an EDB object referring to the datasource named 'Customers'. This database is unsecured with no user or password required.

```
. . .
New EDB DataSource('Customers') NewVar(custdb);
. . .
```

**Example:**

The following script creates an EDB object referring to the datasource named 'Northwind'. The User name and password is supplied.

```
. . .
New EDB DataSource('Northwind') NewVar(northdb) User('MyUsername') Password('MyPass');
. . .
```

**See also**

"Delete" on page 13

### Open

**Format:**

```
Run edbvar.Open
```

The Open method opens the database specified in the EDB object. You must open a database before you can read information from it.

## Close

**Format:**

```
Run edbvar.Close
```

The Close method closes the database specified in the EDB object. You should close a database when you are finished using it.

## Delete

**Format:**

```
Delete Object(edbvar);
```

The Delete method deletes the EDB object. It also closes the database if it is open.

# EQuery Object

The EQuery object represents a search result. You must create an EQuery object using the NEW EQuery command before attempting to search the associated database.

**Table 4: EQuery Methods**

| Method Name  | Method Description                                   |
|--------------|--|
| New          | Creates the EQuery object.                           |
| RunQuery     | Runs a query.  |
| RunSQLTables | Runs a query on the set of database tables available |
| GetNext      | Makes the next record in the result available.       |
| Close        | Closes the query.                                    |
| Delete       | Deletes the Result object.                           |

**Table 5: EQuery Properties**

| Property Name | Data Type | Property Description  |
|---------------|-----------|---|
| Fields        | Group     | The fields in the current search set. You get the information from a field, by using the Field property followed by the name of the field separated by a period. For example, if your result set has a field called Name, you can retrieve this information using the <code>queryvar.Fields.Name</code> syntax. |
| CurrField     | Dependent | This is an alternate way to get field information. Set the <code>CurrFieldName</code> property to the name of the field you wish, then the syntax <code>queryvar.CurrField</code> gives you the value of that field.  |

**Table 5: EQuery Properties**

| Property Name | Data Type | Property Description  |
|---------------|-----------|---|
| CurrFieldName | String    | The name current field. This is used with CurrField above..                               |
| ErrorCode     | Integer   | The FrameScript error code for the last method.   |
| ErrorMsg      | String    | The Text of the last error.   |
| ODBCError     | Integer   | The ODBC error code for the last ODBC call.   |
| QueryRun      | Integer   | True if a query has been run; False otherwise.  |
| AtEOF         | Integer   | True if the current record pointer is at the end of the list of results; False otherwise. |
| Select        | String    | The SQL select string used to run the query.  |
| EDB           | EDB       | The EDB database object associated with this query.                                       |
| NumCols       | Integer   | The number of fields (or columns) in this result set.                                     |
| TableTypes    | String    | The type of tables for the SQLTables query.   |

## Method Descriptions

### New

#### Format:

```
New EQuery EDB(databaseobject) [Select(Selectstr)] [TableTypes(tableStr)]
NewVar (dbvar) ;
```

The New method creates a new EQuery object. You must create an EQuery object before attempting to perform a search or try to access a result set. The parameters are optional. You may set them via properties if you don't specify them on the New command.

**Table 6: NEW EQuery Options**

| Option Name | Option Description   |
|-------------|--|
| EDB         | Identifies the EDB object associated with this query.      |
| Select      | The SQL select command for the search.                     |
| TableTypes  | The list of table types for a SqlTables type query.        |
| NewVar      | The name of the variable to hold the newly created object. |

#### Example:

The following script creates an EQuery object after creating a EDB object for the database.

```
. . .
New EDB DataSource('Customers') NewVar(custdb) User('MyUsername') Password('MyPass');
New EQuery EDB(custdb) Select('Select * From CustTable') NewVar(custquery);
. . .
```

## RunQuery

**Format:**

```
Run queryVar.RunQuery;
```

The RunQuery method executes a query on the specified database and creates a result set. You must have previously specified a Select statement.

## RunSqlTables

**Format:**

```
Run queryVar.RunSqlTables;
```

The RunSqlTables method executes a query on the specified database and creates a result set that includes set of tables in the database. This query allows you to examine the tables and fields in a database.

## GetNext

**Format:**

```
Run queryVar.GetNext;
```

The GetNext method moves the current pointer to the next record in the results set and loads the current fields into the .Fields property of the query object.

## Delete

**Format:**

```
Delete Object(equeryvar);
```

The Delete method deletes the EQuery object.

