



**Visual FoxPro Client-Server Handbook**  
2<sup>nd</sup> Edition

Author: Stamati Crook  
stamati@redware.com  
Date: 8 September 2001  
Version: 4.1  
Document: vfphandbook41.doc

© REDWARE 1996, 2001.

## Shareware Licence

### Copyright © REDWARE 1996, 2001.

8 September 2001 - Version 4.1

All rights reserved. This book is **shareware** and may be downloaded and stored on a single computer for 30 days for the purposes of evaluation only. Registration is required by making the appropriate payment at the **redware** website. The book is copyright and no part shall be reproduced, stored in a retrieval system, or transferred by any means: electronic, mechanical, photocopying, recording, or otherwise without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this handbook, the publisher and author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein. For information, please contact:

**redware research ltd**, 104 Tamworth Road, Hove BN3 5FH, England.  
<http://www.redware.com>

## Acknowledgements

### Second Edition September 2001

Thank you this time to Victor, Phong and especially James for helping build really big database systems during our roller coaster ride at First Telecom. Thanks also to the job market for letting me take a break and update this book.

### First Edition December 1996

Thank you to the technical team at F1 Computing Systems past and present for eight years of implementing FoxPro projects. All of my FoxPro experience has resulted from working with Ian, Phong, James, Danny and David on various projects during my eight years at F1. They still have the best training courses and FoxPro team in the UK. Thank you especially to James Thornton at F1 Computing who put me straight on a few things regarding SQL Server. Any errors remaining in this book are, of course, down to me and I apologise in advance for them. Please email me with your comments, good and bad.

# Contents

---

<b>CONTENTS.....</b>	<b>3</b>
<b>1. OBJECTIVES.....</b>	<b>5</b>
<b>2. TUTORIAL.....</b>	<b>6</b>
<i>Create a Connection.....</i>	<i>6</i>
<i>Create a Remote View .....</i>	<i>6</i>
<i>Updating Data.....</i>	<i>7</i>
<i>Table Buffering.....</i>	<i>7</i>
<i>Using Parameterised Remote Views.....</i>	<i>7</i>
<i>Executing a Stored Procedure.....</i>	<i>8</i>
<i>Creating a client-server Form.....</i>	<i>8</i>
<i>Summary.....</i>	<i>10</i>
<b>3. CONNECTIONS .....</b>	<b>11</b>
ODBC.....	11
<i>ODBC String Connection.....</i>	<i>11</i>
<i>ODBC Data Source Administrator.....</i>	<i>12</i>
<i>ODBC Performance Tips .....</i>	<i>13</i>
CREATE CONNECTION.....	14
CONNECTION DESIGNER.....	14
<i>Data Source.....</i>	<i>15</i>
<i>Data Processing Options.....</i>	<i>15</i>
<i>Asynchronous Connections .....</i>	<i>16</i>
<i>TimeOut Intervals.....</i>	<i>16</i>
<i>Connection Properties.....</i>	<i>16</i>
<b>4. REMOTE VIEWS .....</b>	<b>18</b>
VIEW DESIGNER .....	18
<i>Creating a Remote View.....</i>	<i>18</i>
<i>Fields.....</i>	<i>19</i>
<i>Filter.....</i>	<i>20</i>
<i>Join.....</i>	<i>21</i>
<i>Order By.....</i>	<i>22</i>
<i>Group By .....</i>	<i>22</i>
<i>Update Criteria .....</i>	<i>23</i>
PARAMETERISED VIEWS.....	25
CREATE SQL VIEW .....	26
VIEW PROPERTIES .....	27
OPTIMISING REMOTE VIEWS .....	28
<i>Shared Connections.....</i>	<i>28</i>
<i>Update Criteria .....</i>	<i>28</i>
<i>Advanced Options.....</i>	<i>29</i>
<i>Remote View Properties .....</i>	<i>30</i>
<i>Recommended Settings .....</i>	<i>31</i>
WORKING WITH REMOTE VIEWS .....	31
PARAMETERISED VIEWS.....	32
<b>5. DATA BUFFERING .....</b>	<b>34</b>

SPECIFYING DATA BUFFERING .....	34
SAVING CHANGES .....	35
REVERTING CHANGES .....	36
DETERMINING UPDATES.....	36
ERROR HANDLING.....	38
FORM AND DATA ENVIRONMENT BUFFERING PROPERTIES .....	39
COMMIT AND ROLLBACK .....	40
SETTING DEFAULTS.....	41
OFFLINE VIEWS.....	42
<b>6. SQL PASS THROUGH .....</b>	<b>44</b>
<i>Asynchronous Mode</i> .....	45
<i>Batch Mode</i> .....	45
SQL PASS THROUGH AND DATA BUFFERING.....	46
PREPARING SQL STATEMENTS .....	47
ODBC EXTENSIONS.....	48
STORED PROCEDURES .....	49
<b>7. CLIENT-SERVER APPLICATION DESIGN .....</b>	<b>50</b>
PERFORMANCE BOTTLENECKS .....	50
PARAMETERISED VIEWS.....	50
LOCAL VALIDATIONS .....	51
TRANSACTIONS .....	52
STORED PROCEDURES .....	52
SQL PASS THROUGH.....	52
CONNECTIONS.....	53
MICROSOFT TRANSACTION SERVER .....	53
<b>8. DATABASE MAINTENANCE.....</b>	<b>54</b>
UPSIZING A FOXPRO DATABASE .....	54
<i>Visual FoxPro Upsizing Wizard</i> .....	54
<i>Upsizing Considerations</i> .....	58
DATA MANIPULATION LANGUAGE.....	60
<i>CREATE TABLE</i> .....	61
<i>ALTER TABLE</i> .....	62
<i>CREATE INDEX</i> .....	62
<i>GENDBC.PRG</i> .....	62
<b>9. INDEX .....</b>	<b>63</b>

# 1. Objectives

---

This Handbook aims to cover all the features required to implement client-server database applications using Visual FoxPro as the front end. It is designed as a reference text for experienced programmers and should be used in conjunction with the FoxPro programmers' manual.

An experienced programmer with access to a client-server database should a few days to work through the handbook and discover the implementation tricks and traps of client-server development.

After reading this handbook you will have the knowledge to:

- Configure and optimise shared connections with the database server.
- Create and optimise remote parameterised views.
- Execute SQL pass through onto the database server both synchronously and asynchronously.
- Upsize and modify database definitions on the server from FoxPro.
- Design and optimise a client-server application.

This book was originally written for Visual FoxPro 6.0 and has been updated and tested fully with Visual FoxPro 7.0. New techniques such as accessing ADO recordsets from FoxPro and storing data as XML files is not covered and you should investigate these if you are using VFP 7.0 and need to interface with other application architectures.

This second edition of the Handbook does not cover aspects specific to server-side database issues. All sections are relevant whatever back end database you are using. Please note that we advise you to load up enough test data to stress test your application during the development phase as the performance aspects of client-server are very different from native FoxPro databases.

*REDWARE also publishes the SQL Server Handbook covering all the essential aspects of implementing a database with Microsoft SQL Server.*

Please feed back any comments on the text of the book especially if there are any important points or topics that you feel are missing from the book or any errors you have uncovered. You may contact the author by email on [stamati@redware.com](mailto:stamati@redware.com).

Please look at our web site on [www.redware.com](http://www.redware.com) for information on related products and updates to this handbook.

*Please remember that this book is shareware and register your copy at <http://www.redware.com/register.html>. You will receive our gratitude for rewarding the effort we have put into creating this resource.*

## 2. Tutorial

---

This tutorial section introduces client-server database programming with Visual FoxPro with a minimum of fuss and bother. We explore the major concepts and get to know our way around the PUBS database that ships with SQL Server.

First let us get to grips with some concepts:

- FoxPro uses the Windows ODBC drivers to communicate with the client-server database. You will need to install the appropriate driver on each workstation.
- Remote Views can be created in a database container to reference a client server database query. The retrieved data is represented as a FoxPro cursor.
- Cursors employ table buffering to control the timing of updates sent to the database server.
- Direct communication with the database server is possible using SQL pass through commands.

### Create a Connection

A connection can be created with reference to an ODBC datasource defined on the workstation or directly using a connection string. The following example connects to the PUBS database on a locally installed copy of SQL Server using the system administrator username and password.

```
? SQLSTRINGCONNECT( ;  
  'DRIVER=SQL Server;SERVER=(local);UID=sa;PWD=;DATABASE=pubs')
```

The SQLSTRINGCONNECT() function will return a positive integer referring to the connection handle if successful. Incorrect passwords will usually result in a user prompt while other errors will be trapped by the AERROR() function.

You can test a connection by using SQLTABLES() to get a list of the defined tables returned from the database server into a cursor. The PUBS database should contains several tables including AUTHORS and PUBLISHERS.

```
? SQLTABLES(1, 'table')  
BROWSE
```

We can also execute database commands directly against the database server. The following command will return a read-only copy of the AUTHORS table into a local cursor called fred:

```
? SQLEXEC( 1, 'select * from authors', 'fred' )
```

SQLEXEC returns a 1 if completed successfully, 0 if still processing asynchronously, and -1 if there is an error.

### Create a Remote View

A remote view can be created programmatically but requires a database container to be open for update. Most valid SQL SELECT clauses can be used to define a remote view.

```
CREATE DATABASE dbctutorial  
CREATE CONNECTION conpubs ;  
  CONNSTRING 'DRIVER=SQL Server;SERVER=(local);UID=sa;PWD=;DATABASE=pubs'
```

```
CREATE SQL VIEW vueauthor REMOTE ;
    CONNECTION conpubs SHARED AS select * from authors
SELECT 0
USE dbctutorial!vueauthor
BROWSE
```

### Updating Data

The default definition for a remote view is to create read only data. We must change the view definition by setting the SENDUPDATES parameter to true either in code as shown below or by using the database designer:

```
? DBSETPROP( 'vueauthor', "VIEW", "SENDUPDATES", .t.)
```

Now we can browse and alter data. Notice the error messages from the server if you attempt to set an illegal value (for example by entering letters into the ZIP field).

```
USE dbctutorial!vueauthor
BROWSE
```

### Table Buffering

Table buffering can be useful if you want to update several records in a cursor and control the moment they are updated onto the database. Table buffering is set on an already open cursor using the CURSORSETPROP() function below:

```
SELECT vueauthor
? CURSORSETPROP("Buffering",5)
```

You can now make any changes you require without sending anything to the server. You can cancel all the changes as shown below:

```
REPLACE ALL au_fname WITH 'fred'
? TABLEREVERT(.t.)
```

Or you use the TABLEUPDATE() function to update onto the database server:

```
? TABLEUPDATE(.t., .t.)
```

### Using Parameterised Remote Views

One of the primary design criteria for a scalable client-server application is that only small amounts of data are sent to the workstation at any one time. Parameterised Views allow FoxPro variables to be defined as part of the selection clause so that only the required records are retrieved.

In the example below, a parameterised view is created on the authors table that retrieves only the authors that live in a single state:

```
MODIFY DATABASE dbctutorial
CREATE SQL VIEW vuestateauthors REMOTE CONNECTION conpubs SHARE AS
select * from authors where state = ?lcstate
? DBSETPROP('vuestateauthors','view','sendupdates',.t.)
```

The parameterised view is first opened with the NODATA clause so that no data is retrieved from the server. The required value for the State is specified in the appropriate variable and selecting

the empty cursor and issuing a `REQUERY()` command will interrogate the server and retrieve only the required records.

```
USE vuestateauthors NODATA
lcstate = 'CA'
REQUERY()
lcstate = 'TX'
REQUERY()
```

## Executing a Stored Procedure

Another great advantage of a client-server database is that programs, known as stored procedures, can be defined to run on the server minimising the traffic passing to and fro between server and workstation.

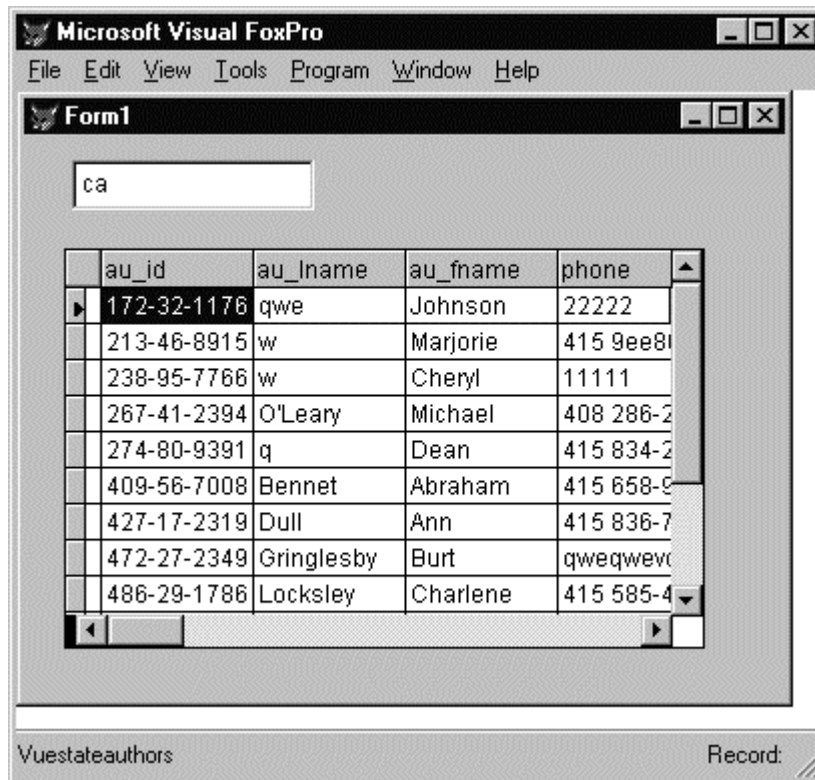
The following example shows how to call the `BYROYALTY` stored procedure which returns a cursor of author identifiers who receive a particular percentage royalty:

```
OPEN DATABASE dbctutorial
lnhandle = SQLCONNECT( 'conpubs' )
? SQLEXP( lnhandle, 'exec byroyalty 40', 'fred' )
```

## Creating a client-server Form

FoxPro forms behave in a similar fashion with client-server remote views as they do with local views and local tables. Care is required to manage the retrieval of the data and updating using table buffering but, otherwise, the behaviours of the cursor is the same in all three environments.

We shall create a form that prompts the user to enter the state required and then retrieves the corresponding authors into a grid for updating. The `VUESTATEAUTHORS` parameterised view is employed for this form, which will operate with record level buffering on the cursor.



First create the form and right click to add the VUESTATEAUTHORS remote view into the data environment. Select the properties for the cursor and set the NODATAONLOAD property to true so that no data is retrieved from the server when the form is opened. Also specify the BUFFERMODEOVERRIDE property to 3 for record level buffering.

Now drag the image of the cursor from the data environment onto the form to create a grid control.

Finally, create a textbox control that allows for a two character string to be entered for the user to specify the state required and add the following code to refresh the parameterised view and the grid:

```
lcstate = THIS.Value  
SELECT vuestateauthors  
=REQUERY ( )  
THIS.Parent.GrdVuestateauthors.Refresh
```

Run the form and enter CA into the textbox. The grid should refresh and allow you to update the author records. Try entering an invalid zip code to check that a response is returned from the server as you move to the next record in the grid.

### Summary

We have covered most of the concepts involved in creating client-server applications with Visual FoxPro in this short tutorial:

- A connection that uses the Windows ODBC/OLEDB middleware to communicate with the client-server database must be defined before FoxPro can communicate with the server.
- Once the connection has been made, commands can be executed directly against the database server to retrieve data or to execute stored procedures.
- A remote view can be defined in a database container to coordinate the retrieval of data from the server into a local cursor. The SENDUPDATES property must be set to allow updates back onto the server controlled by the table buffering properties of the cursor.
- Parameterised views are used to control the retrieval of small sets of data as required by the application. One secret of client-server application design is to break up access to the data into small queries that will allow the application to scale easily to hundreds of users.
- Forms are created in a similar fashion to standard FoxPro data access with attention required for the specification of parameterised views and to the control of the table buffering.

The remainder of the handbook describes each of these areas in detail and offers instruction in the fine tuning and optimisation of FoxPro access to database server.