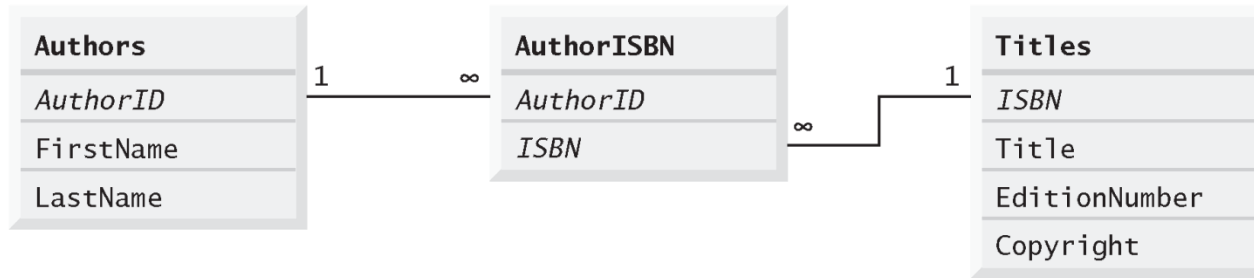
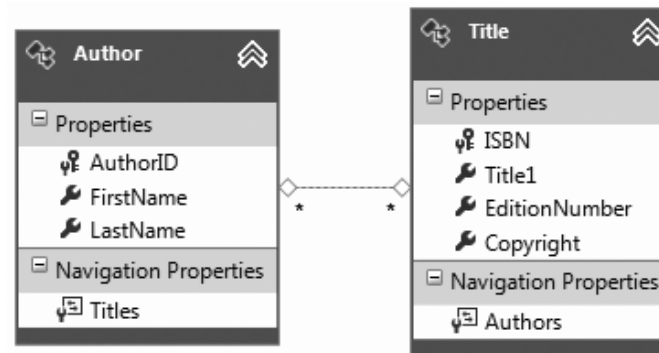




# Databases and LINQ



**Fig. 22.9** | Entity-relationship diagram for the Books database.



**Fig. 22.16** | Entity data model diagram for the **Author** and **Title** entities.



```
1 // Fig. 22.20: DisplayAuthorsTable.cs
2 // Displaying data from a database table in a DataGridView.
3 using System;
4 using System.Data.Entity;
5 using System.Data.Entity.Validation;
6 using System.Linq;
7 using System.Windows.Forms;
8
9 namespace DisplayTable
10 {
11     public partial class DisplayAuthorsTable : Form
12     {
13         // constructor
14         public DisplayAuthorsTable()
15         {
16             InitializeComponent();
17         } // end constructor
18
19         // Entity Framework DbContext
20         private BooksExamples.BooksEntities dbContext =
21             new BooksExamples.BooksEntities();
22     }
```

**Fig. 22.20** | Displaying data from a database table in a DataGridView. (Part I of 4.)



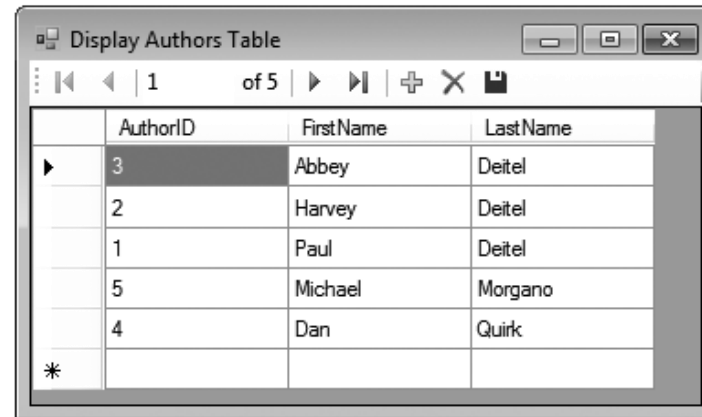
```
23 // load data from database into DataGridView
24 private void DisplayAuthorsTable_Load( object sender, EventArgs e )
25 {
26     // load Authors table ordered by LastName then FirstName
27     dbcontext.Authors
28         .OrderBy( author => author.LastName )
29         .ThenBy( author => author.FirstName )
30         .Load();
31
32     // specify DataSource for authorBindingSource
33     authorBindingSource.DataSource = dbcontext.Authors.Local;
34 } // end method DisplayAuthorsTable_Load
35
36 // click event handler for the Save Button in the
37 // BindingNavigator saves the changes made to the data
38 private void authorBindingNavigatorSaveItem_Click(
39     object sender, EventArgs e )
40 {
41     Validate(); // validate the input fields
42     authorBindingSource.EndEdit(); // complete current edit, if any
43 }
```

**Fig. 22.20** | Displaying data from a database table in a DataGridView. (Part 2 of 4.)



```
44         // try to save changes
45     try
46     {
47         dbcontext.SaveChanges(); // write changes to database file
48     } // end try
49     catch( DbEntityValidationException )
50     {
51         MessageBox.Show( "FirstName and LastName must contain values",
52             "Entity Validation Exception" );
53     } // end catch
54 } // end method authorBindingNavigatorSaveItem_Click
55 } // end class DisplayAuthorsTable
56 } // end namespace DisplayTable
```

**Fig. 22.20 |** Displaying data from a database table in a DataGridView. (Part 3 of 4.)



	AuthorID	FirstName	LastName
▶	3	Abbey	Deitel
	2	Harvey	Deitel
	1	Paul	Deitel
	5	Michael	Morgano
	4	Dan	Quirk
*			

**Fig. 22.20** | Displaying data from a database table in a DataGridView. (Part 4 of 4.)

a) Results of the “All titles” query, which shows the contents of the Titles table ordered by the book titles

	ISBN	Title1	EditionNumber	Copyright
▶	0132121360	Android for Programmers: An App-Driven Approach	1	2012
	013299044X	C How to Program	7	2013
	0133378713	C++ How to Program	9	2014
	0132151006	Internet & World Wide Web How to Program	5	2012
	0132575663	Java How to Program	9	2012
	0132990601	Simply Visual Basic 2010	4	2013
	0133406954	Visual Basic 2012 How to Program	6	2014
	0133379337	Visual C# 2012 How to Program	5	2014
	0136151574	Visual C++ 2008 How to Program	2	2008

All titles

b) Results of the “Titles with 2014 copyright” query

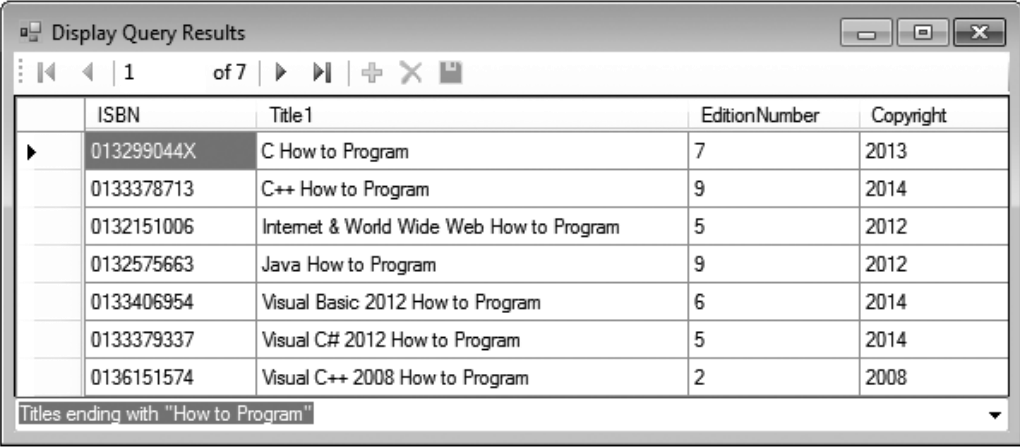
	ISBN	Title1	EditionNumber	Copyright
▶	0133378713	C++ How to Program	9	2014
	0133406954	Visual Basic 2012 How to Program	6	2014
	0133379337	Visual C# 2012 How to Program	5	2014

Titles with 2014 copyright

**Fig. 22.21** | Sample execution of the Display Query Results app. (Part 1 of 2.)



c) Results of the  
“Titles ending  
with 'How to  
Program'” query



The screenshot shows a window titled "Display Query Results" with a table of 7 rows. The first row is highlighted. The table has columns: ISBN, Title1, EditionNumber, and Copyright. Below the table, a text box contains the query: "Titles ending with 'How to Program'".

	ISBN	Title1	EditionNumber	Copyright
▶	013299044X	C How to Program	7	2013
	0133378713	C++ How to Program	9	2014
	0132151006	Internet & World Wide Web How to Program	5	2012
	0132575663	Java How to Program	9	2012
	0133406954	Visual Basic 2012 How to Program	6	2014
	0133379337	Visual C# 2012 How to Program	5	2014
	0136151574	Visual C++ 2008 How to Program	2	2008

Titles ending with "How to Program"

**Fig. 22.21** | Sample execution of the **Display Query Results** app. (Part 2 of 2.)



```
1 // Fig. 22.22: TitleQueries.cs
2 // Displaying the result of a user-selected query in a DataGridView.
3 using System;
4 using System.Data.Entity;
5 using System.Linq;
6 using System.Windows.Forms;
7
8 namespace DisplayQueryResult
9 {
10     public partial class TitleQueries : Form
11     {
12         public TitleQueries()
13         {
14             InitializeComponent();
15         } // end constructor
16
17         // Entity Framework DbContext
18         private BooksExamples.BooksEntities dbcontext =
19             new BooksExamples.BooksEntities();
20     }
```

**Fig. 22.22** | Displaying the result of a user-selected query in a DataGridView.  
(Part I of 3.)



```
21 // load data from database into DataGridView
22 private void TitleQueries_Load( object sender, EventArgs e )
23 {
24     dbcontext.Titles.Load(); // load Titles table into memory
25
26     // set the ComboBox to show the default query that
27     // selects all books from the Titles table
28     queriesComboBox.SelectedIndex = 0;
29 } // end method TitleQueries_Load
30
31 // loads data into titleBindingSource based on user-selected query
32 private void queriesComboBox_SelectedIndexChanged(
33     object sender, EventArgs e )
34 {
35     // set the data displayed according to what is selected
36     switch ( queriesComboBox.SelectedIndex )
37     {
38         case 0: // all titles
39             // use LINQ to order the books by title
40             titleBindingSource.DataSource =
41                 dbcontext.Titles.Local.OrderBy( book => book.Title1 );
42             break;
```

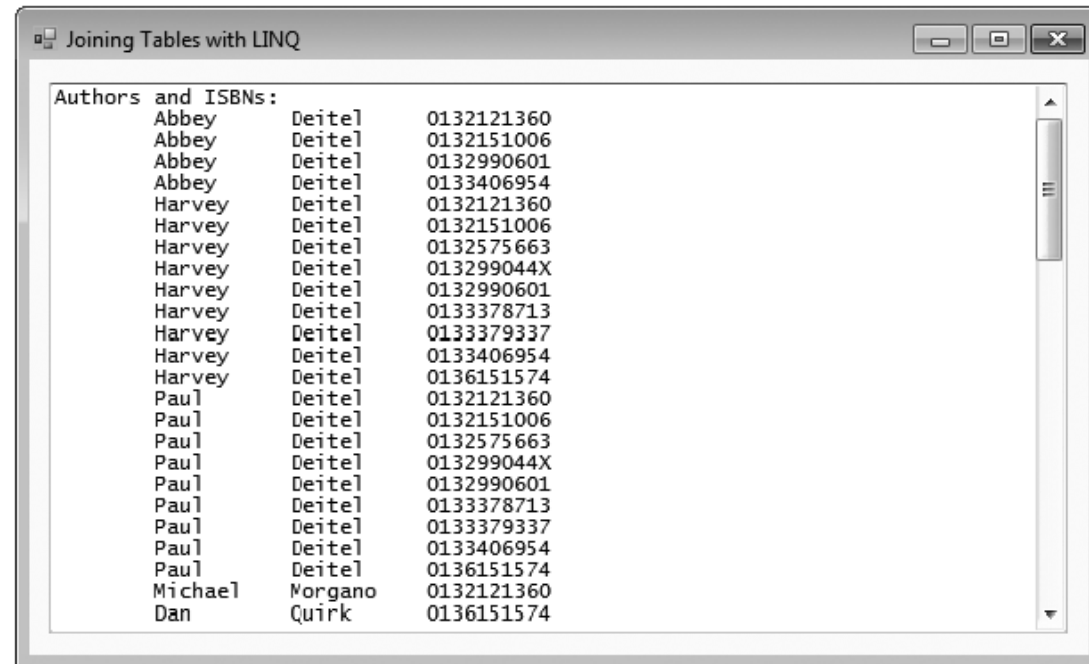
**Fig. 22.22** | Displaying the result of a user-selected query in a DataGridView.  
(Part 2 of 3.)



```
43         case 1: // titles with 2014 copyright
44             // use LINQ to get titles with 2014
45             // copyright and sort them by title
46             titleBindingSource.DataSource =
47                 dbcontext.Titles.Local
48                 .Where( book => book.Copyright == "2014" )
49                 .OrderBy( book => book.Title1 );
50             break;
51         case 2: // titles ending with "How to Program"
52             // use LINQ to get titles ending with
53             // "How to Program" and sort them by title
54             titleBindingSource.DataSource =
55                 dbcontext.Titles.Local
56                 .Where( book =>
57                     book.Title1.EndsWith( "How to Program" ) )
58                 .OrderBy( book => book.Title1 );
59             break;
60     } // end switch
61
62     titleBindingSource.MoveFirst(); // move to first entry
63 } // end method queriesComboBox_SelectedIndexChanged
64 } // end class TitleQueries
65 } // end namespace DisplayQueryResult
```

**Fig. 22.22** | Displaying the result of a user-selected query in a DataGridView.  
(Part 3 of 3.)

a) List of authors and the ISBNs of the books they've authored; sort the authors by last name then first name



Authors and ISBNs:		
Abbey	Deitel	0132121360
Abbey	Deitel	0132151006
Abbey	Deitel	0132990601
Abbey	Deitel	0133406954
Harvey	Deitel	0132121360
Harvey	Deitel	0132151006
Harvey	Deitel	0132575663
Harvey	Deitel	013299044X
Harvey	Deitel	0132990601
Harvey	Deitel	0133378713
Harvey	Deitel	0133379337
Harvey	Deitel	0133406954
Harvey	Deitel	0136151574
Paul	Deitel	0132121360
Paul	Deitel	0132151006
Paul	Deitel	0132575663
Paul	Deitel	013299044X
Paul	Deitel	0132990601
Paul	Deitel	0133378713
Paul	Deitel	0133379337
Paul	Deitel	0133406954
Paul	Deitel	0136151574
Michael	Morgano	0132121360
Dan	Quirk	0136151574

**Fig. 22.23** | Outputs from the Joining Tables with LINQ app. (Part I of 3.)

b) List of authors and the titles of the book's they've authored; sort the authors by last name then first name; for a given author, sort the titles alphabetically

Authors and titles:		
Abbey	Deitel	Android for Programmers: An App-Driven Approach
Abbey	Deitel	Internet & World Wide Web How to Program
Abbey	Deitel	Simply Visual Basic 2010
Abbey	Deitel	Visual Basic 2012 How to Program
Harvey	Deitel	Android for Programmers: An App-Driven Approach
Harvey	Deitel	C How to Program
Harvey	Deitel	C++ How to Program
Harvey	Deitel	Internet & World Wide Web How to Program
Harvey	Deitel	Java How to Program
Harvey	Deitel	Simply Visual Basic 2010
Harvey	Deitel	Visual Basic 2012 How to Program
Harvey	Deitel	Visual C# 2012 How to Program
Harvey	Deitel	Visual C++ 2008 How to Program
Paul	Deitel	Android for Programmers: An App-Driven Approach
Paul	Deitel	C How to Program
Paul	Deitel	C++ How to Program
Paul	Deitel	Internet & World Wide Web How to Program
Paul	Deitel	Java How to Program
Paul	Deitel	Simply Visual Basic 2010
Paul	Deitel	Visual Basic 2012 How to Program
Paul	Deitel	Visual C# 2012 How to Program
Paul	Deitel	Visual C++ 2008 How to Program
Michael	Morgano	Android for Programmers: An App-Driven Approach
Dan	Quirk	Visual C++ 2008 How to Program

**Fig. 22.23** | Outputs from the Joining Tables with LINQ app. (Part 2 of 3.)

c) List of titles grouped by author; sort the authors by last name then first name; for a given author, sort the titles alphabetically

```
Joining Tables with LINQ

Titles grouped by author:
  Abbey Deitel:
    Android for Programmers: An App-Driven Approach
    Internet & World Wide Web How to Program
    Simply Visual Basic 2010
    Visual Basic 2012 How to Program
  Harvey Deitel:
    Android for Programmers: An App-Driven Approach
    C How to Program
    C++ How to Program
    Internet & World Wide Web How to Program
    Java How to Program
    Simply Visual Basic 2010
    Visual Basic 2012 How to Program
    Visual C# 2012 How to Program
    Visual C++ 2008 How to Program
  Paul Deitel:
    Android for Programmers: An App-Driven Approach
    C How to Program
    C++ How to Program
    Internet & World Wide Web How to Program
    Java How to Program
    Simply Visual Basic 2010
    Visual Basic 2012 How to Program
    Visual C# 2012 How to Program
    Visual C++ 2008 How to Program
  Michael Morgano:
    Android for Programmers: An App-Driven Approach
  Dan Quirk:
    Visual C++ 2008 How to Program
```

**Fig. 22.23** | Outputs from the **Joining Tables with LINQ** app. (Part 3 of 3.)



```
1 // Fig. 22.24: JoiningTableData.cs
2 // Using LINQ to perform a join and aggregate data across tables.
3 using System;
4 using System.Linq;
5 using System.Windows.Forms;
6
7 namespace JoinQueries
8 {
9     public partial class JoiningTableData : Form
10    {
11        public JoiningTableData()
12        {
13            InitializeComponent();
14        } // end constructor
15
16        private void JoiningTableData_Load(object sender, EventArgs e)
17        {
18            // Entity Framework DbContext
19            BooksExamples.BooksEntities dbcontext =
20                new BooksExamples.BooksEntities();
21        }
```

**Fig. 22.24** | Creating the BooksDataContext for querying the Books database.





```
22 // get authors and ISBNs of each book they co-authored
23 var authorsAndISBNs =
24     from author in dbcontext.Authors
25     from book in authorTitles
26     orderby author.LastName, author.FirstName
27     select new { author.FirstName, author.LastName, book.ISBN };
28
29 outputTextBox.AppendText( "Authors and ISBNs:" );
30
31 // display authors and ISBNs in tabular format
32 foreach ( var element in authorsAndISBNs )
33 {
34     outputTextBox.AppendText(
35         String.Format( "\r\n\t{0,-10} {1,-10} {2,-10}",
36             element.FirstName, element.LastName, element.ISBN ) );
37 } // end foreach
38
```

**Fig. 22.25** | Getting a list of authors and the ISBNs of the books they've authored.



```
39 // get authors and titles of each book they co-authored
40 var authorsAndTitles =
41     from book in dbcontext.Titles
42     from author in book.Authors
43     orderby author.LastName, author.FirstName, book.Title1
44     select new { author.FirstName, author.LastName,
45                 book.Title1 };
46
47 outputTextBox.AppendText( "\r\n\r\nAuthors and titles:" );
48
49 // display authors and titles in tabular format
50 foreach ( var element in authorsAndTitles )
51 {
52     outputTextBox.AppendText(
53         String.Format( "\r\n\t{0,-10} {1,-10} {2}",
54             element.FirstName, element.LastName, element.Title1 ) );
55 } // end foreach
56
```

**Fig. 22.26** | Getting a list of authors and the titles of the books they've authored.



```
57 // get authors and titles of each book
58 // they co-authored; group by author
59 var titlesByAuthor =
60     from author in dbcontext.Authors
61     orderby author.LastName, author.FirstName
62     select new { Name = author.FirstName + " " + author.LastName,
63         Titles =
64             from book in author.Titles
65             orderby book.Title1
66             select book.Title1 };
67
68 outputTextBox.AppendText( "\r\n\r\nTitles grouped by author:" );
69
```

**Fig. 22.27** | Getting a list of titles grouped by authors. (Part 1 of 2.)



```
70         // display titles written by each author, grouped by author
71         foreach ( var author in titlesByAuthor )
72         {
73             // display author's name
74             outputTextBox.AppendText( "\r\n\t" + author.Name + ":" );
75
76             // display titles written by that author
77             foreach ( var title in author.Titles )
78             {
79                 outputTextBox.AppendText( "\r\n\t\t" + title );
80             } // end inner foreach
81         } // end outer foreach
82     } // end method JoiningTableData_Load
83 } // end class JoiningTableData
84 } // end namespace JoinQueries
```

**Fig. 22.27** | Getting a list of titles grouped by authors. (Part 2 of 2.)

- a) Use the BindingNavigator's controls to navigate through the contacts in the database

Address Book

1 of 6

Address ID: 2

First Name: Lisa

Last Name: Black

Email: lblack@deitel.com

Phone Number: 555-654-3210

Find an entry by last name

Last Name:  Find

Browse All Entries

**Fig. 22.32** | Manipulating an address book. (Part 1 of 3.)

b) Type a search string in the **Last Name:** TextBox then press **Find** to locate contacts whose last names begin with that string; only two names start with “Br” so the BindingNavigator indicates two matching records

Displaying the first of two matching contacts for the current search

**Fig. 22.32** | Manipulating an address book. (Part 2 of 3.)

- c) Click **Browse All Entries** to clear the search string and allow browsing of all contacts in the database

You can now  
browse through all  
six contacts

**Fig. 22.32** | Manipulating an address book. (Part 3 of 3.)



```
1 // Fig. 22.33: Contact.cs
2 // Manipulating an address book.
3 using System;
4 using System.Data;
5 using System.Data.Entity;
6 using System.Data.Entity.Validation;
7 using System.Linq;
8 using System.Windows.Forms;
9
10 namespace AddressBook
11 {
12     public partial class Contacts : Form
13     {
14         public Contacts()
15         {
16             InitializeComponent();
17         } // end constructor
18
19         // Entity Framework DbContext
20         private AddressExample.AddressBookEntities dbcontext = null;
21     }
```

**Fig. 22.33** | Creating the BooksDataContext and defining method RefreshContacts for use in other methods. (Part 1 of 2.)





```
22 // fill our addressBindingSource with all rows, ordered by name
23 private void RefreshContacts()
24 {
25     // Dispose old DbContext, if any
26     if ( dbContext != null )
27         dbContext.Dispose();
28
29     // create new DbContext so we can reorder records based on edits
30     dbContext = new AddressExample.AddressBookEntities();
31
32     // use LINQ to order the Addresses table contents
33     // by last name, then first name
34     dbContext.Addresses
35         .OrderBy( entry => entry.LastName )
36         .ThenBy( entry => entry.FirstName )
37         .Load();
38
39     // specify DataSource for addressBindingSource
40     addressBindingSource.DataSource = dbContext.Addresses.Local;
41     addressBindingSource.MoveFirst(); // go to first result
42     findTextBox.Clear(); // clear the Find TextBox
43 } // end method RefreshContacts
44
```

**Fig. 22.33** | Creating the BooksDataContext and defining method RefreshContacts for use in other methods. (Part 2 of 2.)



```
45 // when the form loads, fill it with data from the database
46 private void Contacts_Load( object sender, EventArgs e )
47 {
48     RefreshContacts(); // fill binding with data from database
49 } // end method Contacts_Load
50
```

**Fig. 22.34** | Calling RefreshContacts to fill the TextBoxes when the app loads.



```
51 // Click event handler for the Save Button in the
52 // BindingNavigator saves the changes made to the data
53 private void addressBindingNavigatorSaveItem_Click(
54     object sender, EventArgs e )
55 {
56     Validate(); // validate input fields
57     addressBindingSource.EndEdit(); // complete current edit, if any
58
59     // try to save changes
60     try
61     {
62         dbContext.SaveChanges(); // write changes to database file
63     } // end try
64     catch ( DbEntityValidationException )
65     {
66         MessageBox.Show( "Columns cannot be empty",
67             "Entity Validation Exception" );
68     } // end catch
69
70     RefreshContacts(); // change back to initial unfiltered data
71 } // end method addressBindingNavigatorSaveItem_Click
72
```

**Fig. 22.35 |** Saving changes to the database when the user clicks Save Data.



```
73 // use LINQ to create a data source that contains only people
74 // with last names that start with the specified text
75 private void findButton_Click( object sender, EventArgs e )
76 {
77     // use LINQ to filter contacts with last names that
78     // start with findTextBox contents
79     var lastNameQuery =
80         from address in dbcontext.Addresses
81         where address.LastName.StartsWith( findTextBox.Text )
82         orderby address.LastName, address.FirstName
83         select address;
84
85     // display matching contacts
86     addressBindingSource.DataSource = lastNameQuery.ToList();
87     addressBindingSource.MoveFirst(); // go to first result
88
89     // don't allow add/delete when contacts are filtered
90     bindingNavigatorAddNewItem.Enabled = false;
91     bindingNavigatorDeleteItem.Enabled = false;
92 } // end method findButton_Click
93
```

**Fig. 22.36** | Finding the contacts whose last names begin with a specified String.



```
94      // reload addressBindingSource with all rows
95      private void browseAllButton_Click( object sender, EventArgs e )
96      {
97          // allow add/delete when contacts are not filtered
98          bindingNavigatorAddNewItem.Enabled = true;
99          bindingNavigatorDeleteItem.Enabled = true;
100         RefreshContacts(); // change back to initial unfiltered data
101     } // end method browseButton_Click
102 } // end class Contacts
103 } // end namespace AddressBook
```

**Fig. 22.37** | Allowing the user to browse all contacts.