

## EXAMPLE - 02

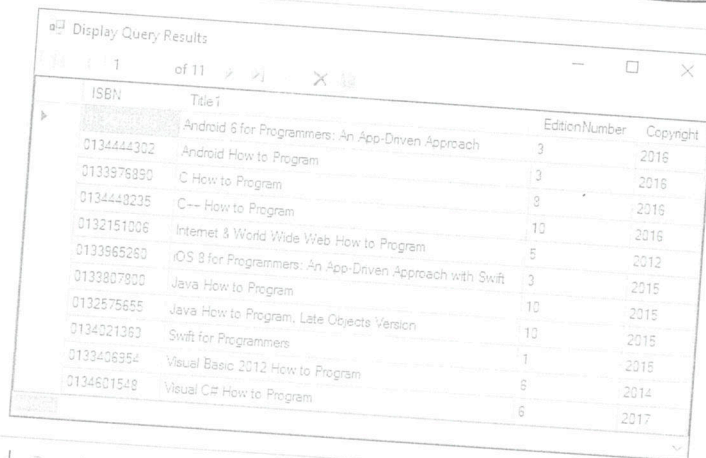
42 calls `EndEdit` on the `authorBindingSource`, which forces it to save any pending changes into the `BooksEntities` model in memory. Finally, line 47 calls `SaveChanges` on the `BooksEntities` object `dbContext` to store any changes into the database. We placed this call in a try statement, because the `Authors` table does not allow empty values for the first name and last name—these rules were configured when we originally created the database. When `SaveChanges` is called, any changes stored into the `Authors` table must satisfy the table's rules. If any do not, a `DbEntityValidationException` occurs.

## EXAMPLE 02

### 22.6 Dynamically Binding Query Results

Next we show how to perform several different queries and display the results in a `DataGridView`. This app only reads data from the entity data model, so we disabled the buttons in the `BindingNavigator` that enable the user to add and delete records—simply select each button and set its `Enabled` property to `False` in the `Properties` window. You also could delete these buttons from the `BindingNavigator`. Later, we'll explain why we do not support modifying the database in this example.

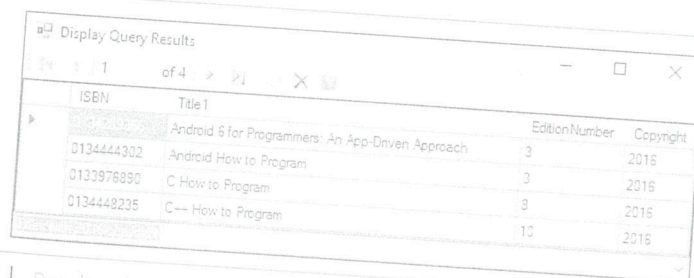
The `Display Query Results` app (Figs. 22.23–22.25) allows the user to select a query from the `ComboBox` at the bottom of the window, then displays the results of the query.



The screenshot shows a window titled "Display Query Results" with a toolbar at the top. Below the toolbar is a table with four columns: ISBN, Title1, EditionNumber, and Copyright. The table contains 11 rows of data, representing books from the Titles table ordered by title.

ISBN	Title1	EditionNumber	Copyright
0134444302	Android 6 for Programmers: An App-Driven Approach	3	2016
0133976890	Android How to Program	3	2016
0134448235	C How to Program	8	2016
0132151006	C++ How to Program	10	2016
0133665260	Internet & World Wide Web How to Program	5	2012
0133807800	iOS 8 for Programmers: An App-Driven Approach with Swift	3	2015
0132575655	Java How to Program	10	2015
0134021363	Java How to Program, Late Objects Version	10	2015
0133406954	Swift for Programmers	1	2015
0134406954	Visual Basic 2012 How to Program	6	2014
0134601548	Visual C# How to Program	6	2017

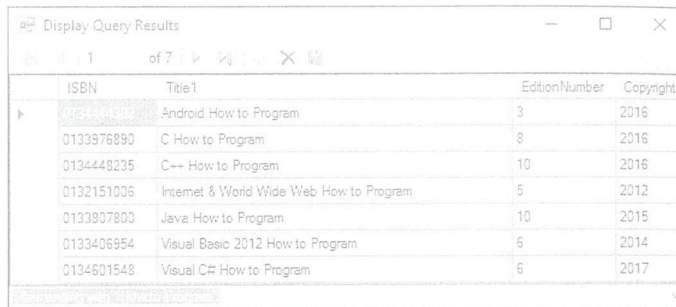
Fig. 22.23 | Results of the `Display Query Results` app's `All titles` query, which shows the contents of the `Titles` table ordered by the book titles.



The screenshot shows the same "Display Query Results" window, but now displaying only 4 rows of data, representing books with a 2016 copyright year.

ISBN	Title1	EditionNumber	Copyright
0134444302	Android 6 for Programmers: An App-Driven Approach	3	2016
0133976890	Android How to Program	3	2016
0134448235	C How to Program	8	2016
0132151006	C++ How to Program	10	2016

Fig. 22.24 | Results of the `Display Query Results` app's `Titles with 2016 copyright` query.



ISBN	Title1	EditionNumber	Copyright
0133976890	Android How to Program	3	2016
0134449235	C How to Program	8	2016
0132151006	C++ How to Program	10	2016
0133807800	Internet & World Wide Web How to Program	5	2012
0133406954	Java How to Program	10	2015
0134601548	Visual Basic 2012 How to Program	6	2014
	Visual C# How to Program	6	2017

Fig. 22.25 | Results of the Display Query Results app's Titles ending with "How to Program" query.

### 22.6.1 Creating the Display Query Results GUI

Perform the following steps to build the Display Query Results app's GUI.

#### Step 1: Creating the Project

Perform the steps in Section 22.5.2 to create a new Windows Forms Application project named `DisplayQueryResult` in the same solution as the `DisplayTable` app. Rename the `Form1.cs` source file to `TitleQueries.cs`. Set the Form's Text property to Display Query Results. Be sure to add references to the `BooksExamples` and `EntityFramework` libraries, add the connection string to the project's `App.Config` file and set the `DisplayQueryResult` project as the startup project.

#### Step 2: Creating a DataGridView to Display the Titles Table

Follow Steps 1 and 2 in Section 22.5.3 to create the data source and the `DataGridView`. For this example, select the `Title` class (rather than `Author`) as the data source, and drag the `Title` node from the Data Sources window onto the form. Remove the `Authors` column from the `DataGridView`, as it will not be used in this example.

#### Step 3: Adding a ComboBox to the Form

In Design view, add a `ComboBox` named `queriesComboBox` below the `DataGridView` on the Form. Users will select which query to execute from this control. Set the `ComboBox`'s Dock property to Bottom and the `DataGridView`'s Dock property to Fill.

Next, you'll add the names of the queries to the `ComboBox`. Open the `ComboBox`'s String Collection Editor by right clicking the `ComboBox` and selecting `Edit Items....` You also can access the String Collection Editor from the `ComboBox`'s smart tag menu. A smart tag menu provides you with quick access to common properties you might set for a control (such as the Multiline property of a `TextBox`), so you can set these properties directly in Design view, rather than in the Properties window. You can open a control's smart tag menu by clicking the small arrowhead



that appears in the control's upper-right corner in Design view when the control is selected. In the String Collection Editor, add the following three items to `queriesComboBox`—one for each of the queries we'll create:

1. All titles
2. Titles with 2016
3. Titles ending with

### 22.6.2 Coding the Display Query Results GUI

Next you'll create the code

```

1 // Fig. 22.26: TitleQueries.cs
2 // Displaying the titles
3 using System;
4 using System.Data;
5 using System.Linq;
6 using System.Windows.Forms;
7
8 namespace DisplayQueryResults
9 {
10     public partial class TitleQueries : Form
11     {
12         public TitleQueries()
13         {
14             InitializeComponent();
15         }
16
17         // Entity Framework connection string
18         private string connectionString = "Data Source=BooksExamples.db;Initial Catalog=BooksExamples;Integrated Security=True;MultipleActiveResultSets=True;User=sa;Password=sa;";
19
20         // load the data
21         private void LoadData()
22         {
23             dbcon = new SqlConnection(connectionString);
24             dbcon.Open();
25
26             // select all titles
27             string query = "SELECT * FROM Title";
28
29             // load the data
30             private void LoadData()
31             {
32                 object[] titles = dbcon.Query(query).ToArray();
33                 titlesList.AddRange(titles);
34             }
35
36             // switch on the selected query
37             switch (queriesComboBox.SelectedIndex)
38             {
39                 case 0:
40                     LoadData();
41                     dataGridView1.DataSource = titlesList;
42                     break;

```

Fig. 22.26 | Display Query Results GUI



1. All titles
2. Titles with 2016 copyright
3. Titles ending with "How to Program"

### 22.6.2 Coding the Display Query Results App

Next you'll create the code for this app (Fig. 22.26).

```

1 // Fig. 22.26: 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         }
16
17         // Entity Framework DbContext
18         private BooksExamples.BooksEntities dbContext =
19             new BooksExamples.BooksEntities();
20
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         }
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. 22.26 | Displaying the result of a user-selected query in a DataGridView. (Part 1 of 2.)

Number	Copyright
2016	2016
2016	2016
2016	2016
2012	2015
2015	2014
2017	

binding with "How to

pp's GUI.

Forms Application project  
ayTable app. Rename the  
roperty to Display Query  
ntityFramework libraries,  
d set the DisplayQuery-

ble  
e and the DataGridView.  
the data source, and drag  
move the Authors column

r the DataGridView on the  
l. Set the ComboBox's Dock  
ill.

Open the ComboBox's String  
Edit Items.... You also can  
r menu. A smart tag menu  
it set for a control (such as  
operties directly in Design  
ontrol's smart tag menu by

when the control is select-  
ns to queriesComboBox—