

CSHARP CLASS II DATABASE CONNECTIONS

KEVIN.LI@GEORGIANCOLLEGE.CA

OUTLINE

- ADO.net
- WinForm sample
- MVC sample

.NET DATA PROVIDER CORE OBJECTS

C17, Slide 3

| Object | Description |
|--------------|--|
| Connection | Establishes a connection to a database. |
| Command | Represents an individual SQL statement that can be executed against the database. |
| Data reader | Provides read-only, forward-only access to the data in a database. |
| Data adapter | Provides the link between the command and connection objects and a dataset object. |

DATA PROVIDERS INCLUDED WITH THE .NET FRAMEWORK

| Provider | Namespace | Lets you access... |
|------------|------------------------------------|-----------------------------------|
| SQL Server | <code>System.Data.SqlClient</code> | SQL Server databases |
| OLE DB | <code>System.Data.OleDb</code> | Any database that supports OLE DB |
| ODBC | <code>System.Data.Odbc</code> | Any database that supports ODBC |

CLASS NAMES FOR THE DATA PROVIDERS

C17, Slide 5

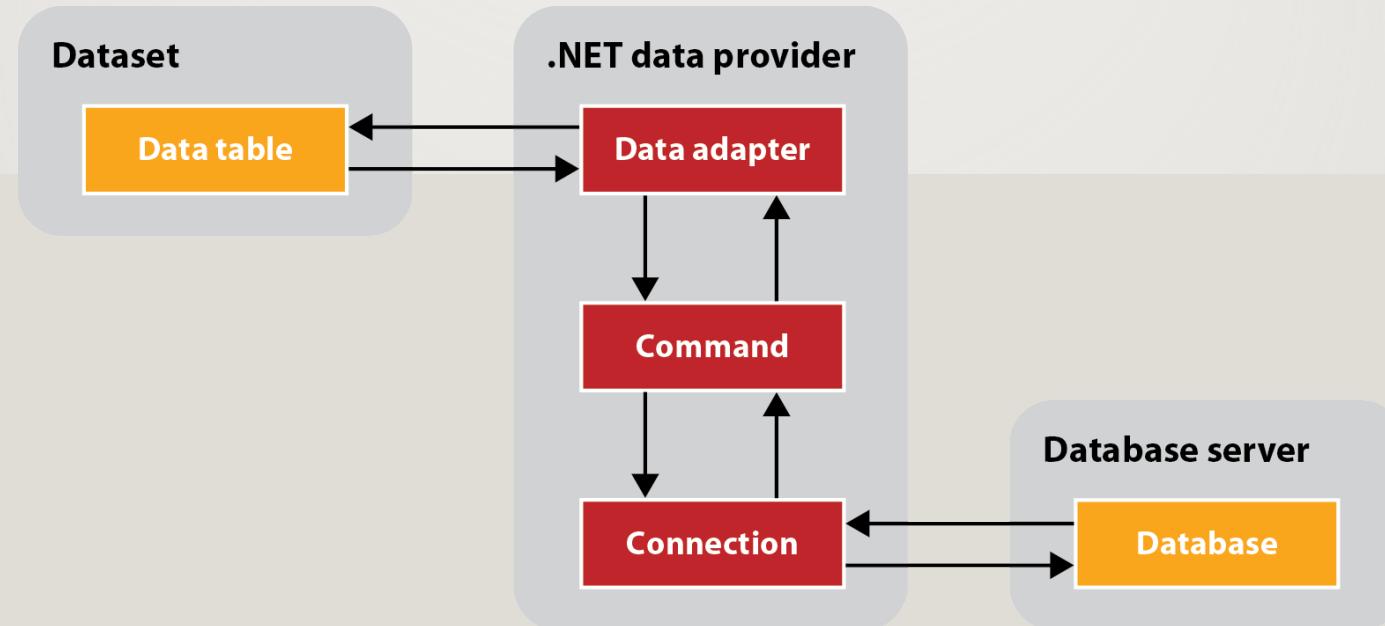
| Object | SQL Server | OLE DB | ODBC |
|--------------|-----------------------------|-------------------------------|------------------------------|
| Connection | <code>SqlConnection</code> | <code>OleDbConnection</code> | <code>OdbcConnection</code> |
| Command | <code>SqlCommand</code> | <code>OleDbCommand</code> | <code>OdbcCommand</code> |
| Data reader | <code>SqlDataReader</code> | <code>OleDbDataReader</code> | <code>OdbcDataReader</code> |
| Data adapter | <code>SqlDataAdapter</code> | <code>OleDbDataAdapter</code> | <code>OdbcDataAdapter</code> |

A using directive for the SQL Server data provider namespace

```
using System.Data.SqlClient;
```

BASIC ADO.NET COMPONENTS

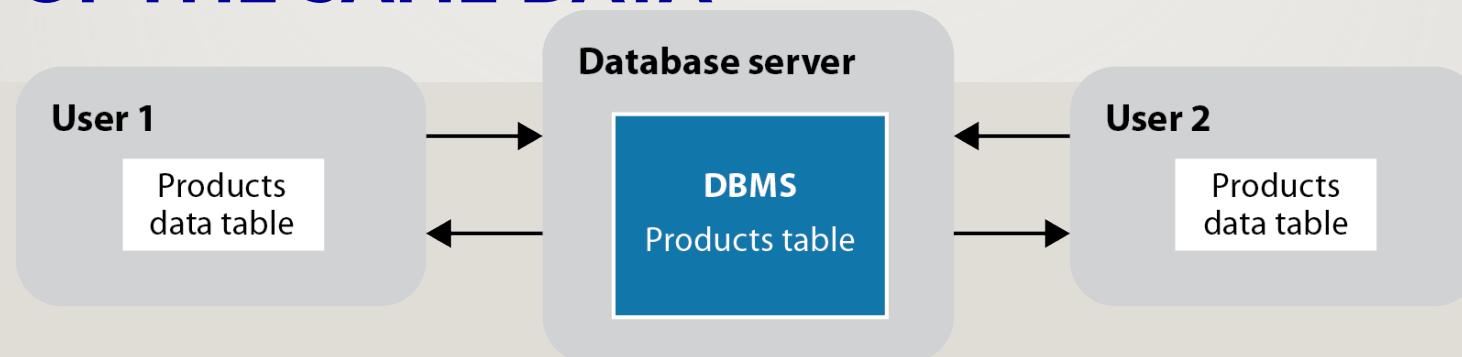
C17, Slide 6



ADO stands for ActiveX Data Objects.

ADO.NET is a database technology of .NET Framework used to connect application system and database server.

TWO USERS WHO ARE WORKING WITH COPIES OF THE SAME DATA



CONCURRENCY CONCEPTS

C17, Slide 8

- When two or more users retrieve the data in the same row of a database table at the same time, it is called *concurrency*. Because ADO.NET uses a disconnected data architecture, the database management system can't prevent this from happening.
- If two users try to update the same row in a database table at the same time, the second user's changes could overwrite the changes made by the first user. Whether or not that happens, though, depends on the *concurrency control* that the programs use.
- By default, ADO.NET uses *optimistic concurrency*. This means that the program checks to see whether the database row that's going to be updated or deleted has been changed since it was retrieved. If it has, a *concurrency exception* occurs and the update or deletion is refused. Then, the program should handle the exception.

CONCURRENCY CONCEPTS (CONT.)

C17, Slide 9

- If optimistic concurrency isn't in effect, the program doesn't check to see whether a row has been changed before an update or deletion takes place. Instead, the operation proceeds without throwing an exception. This is referred to as "*last in wins*" because the last update overwrites any previous update. And this can lead to errors in the database.

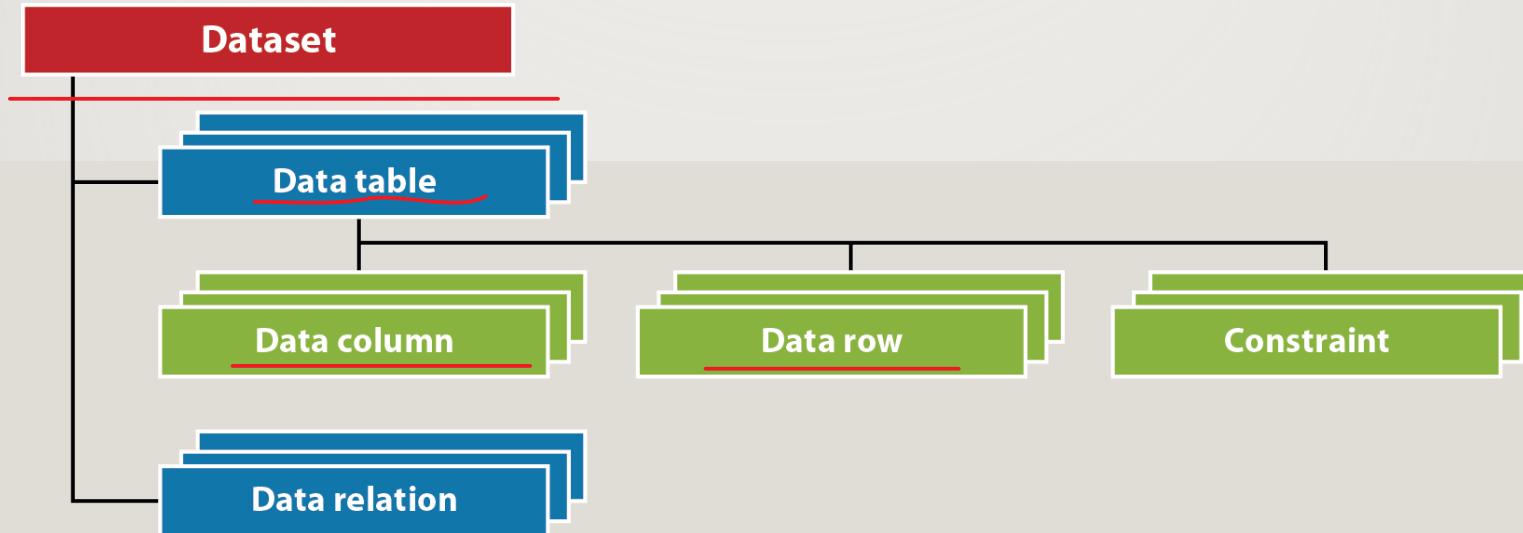
HOW TO AVOID CONCURRENCY ERRORS

C17, Slide 10

- For many applications, concurrency errors **rarely occur**. As a result, optimistic concurrency is adequate.
- If concurrency is likely to be a problem, a program that uses a dataset can be designed so it **updates the database and refreshes the dataset frequently**. That way, concurrency errors are less likely to occur.
- Another alternative is to design a program so it retrieves and **updates just one row at a time**. That way, there's less chance that two users will retrieve and update the same row at the same time.

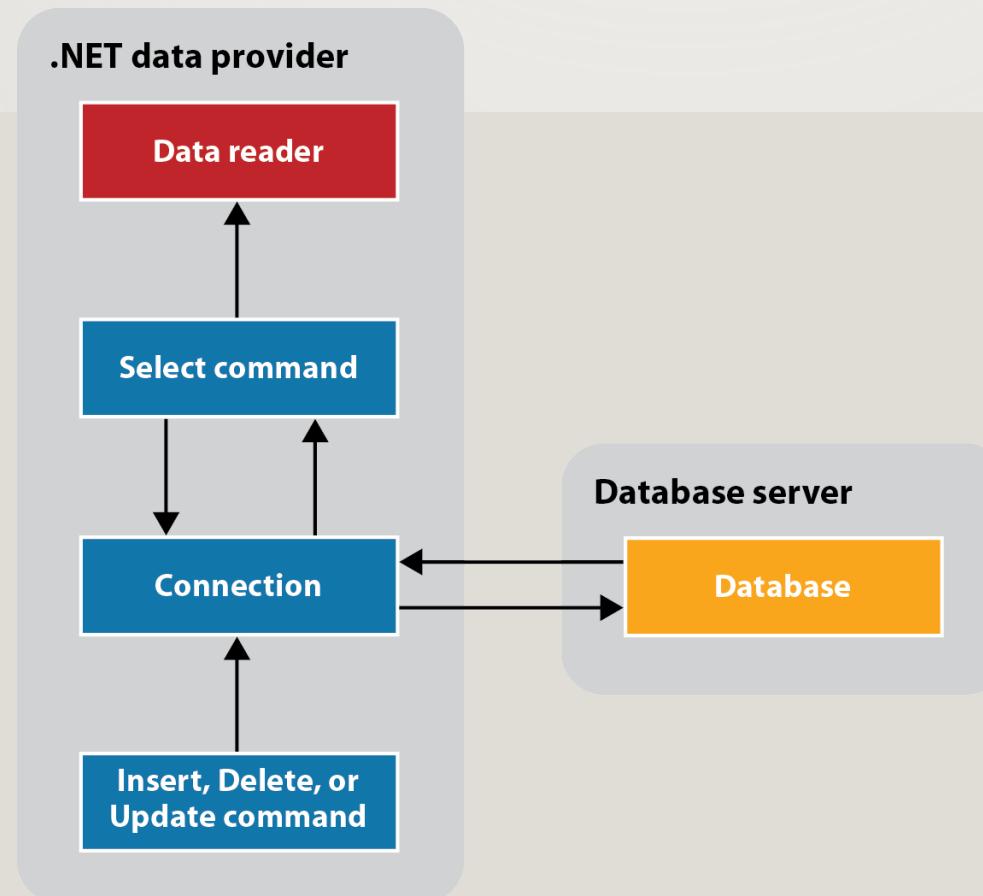
THE BASIC DATASET OBJECT HIERARCHY

C17, Slide 11

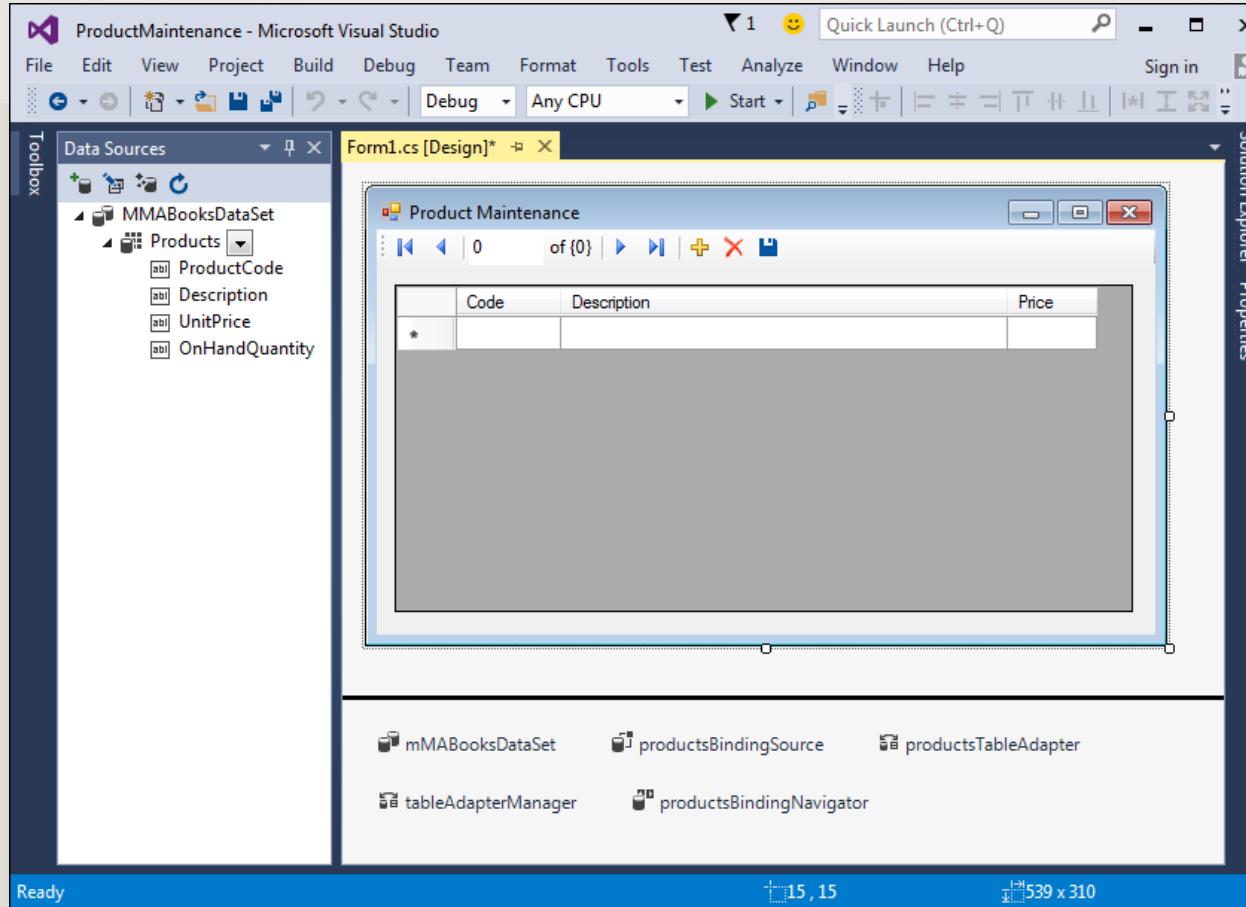


ADO.NET COMPONENTS FOR ACCESSING A DATABASE DIRECTLY

C17, Slide 12



ADO.NET OBJECTS CREATED USING THE DATA SOURCES WINDOW



ADO.NET OBJECTS CREATED USING CODE

C17, Slide 14

```
string connectionString =
    "Data Source=localhost\\SqlExpress;Initial Catalog=MMABooks;" +
    "Integrated Security=True";
SqlConnection connection = new SqlConnection(connectionString);

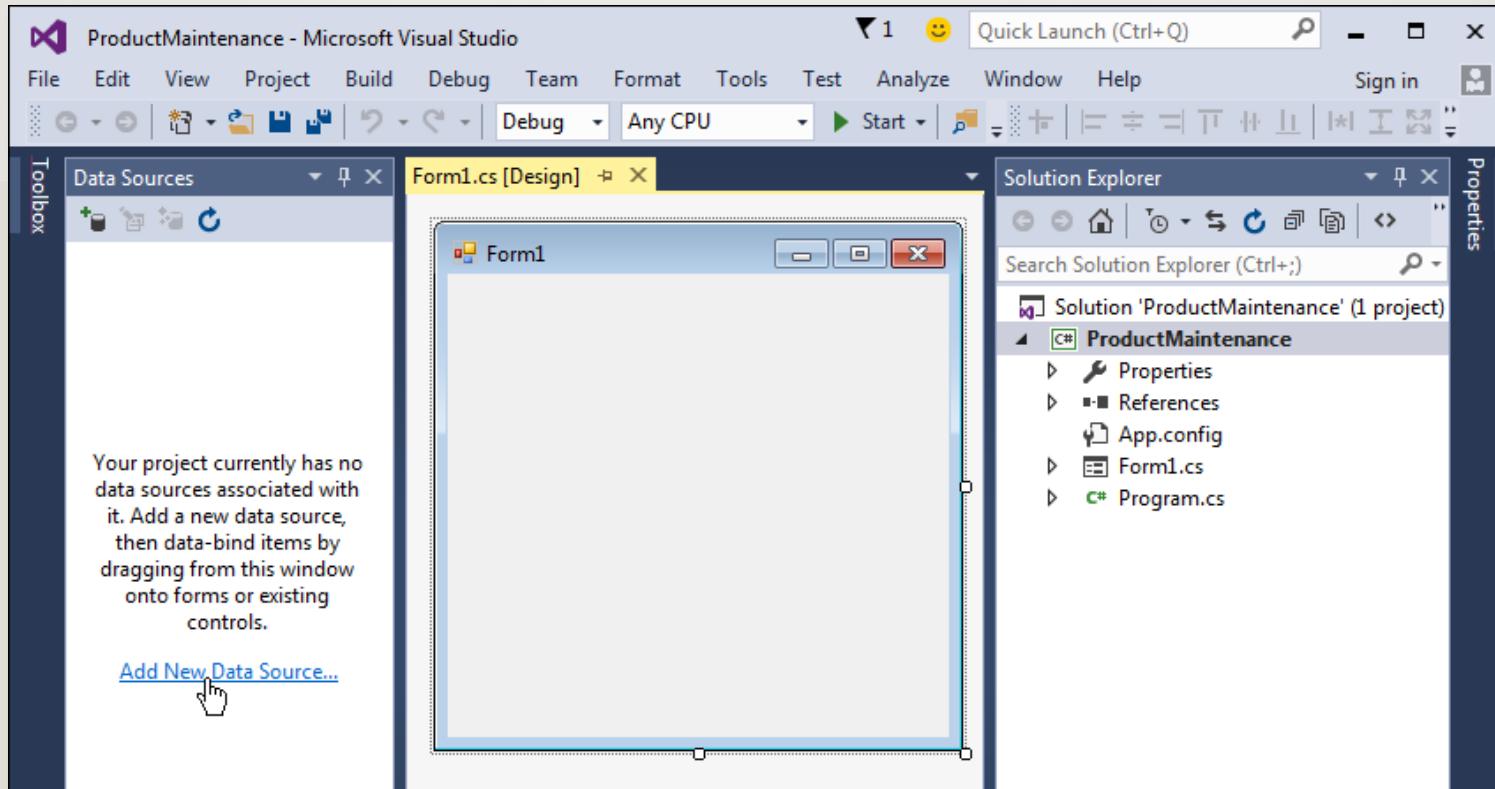
string selectStatement =
    "SELECT * FROM Products ORDER BY ProductCode";
SqlCommand selectCommand =
    new SqlCommand(selectStatement, connection);

SqlDataAdapter productsDataAdapter =
    new SqlDataAdapter(selectCommand);

DataSet productsDataSet = new DataSet();
productsDataSet = productDataAdapter.Fill()
```

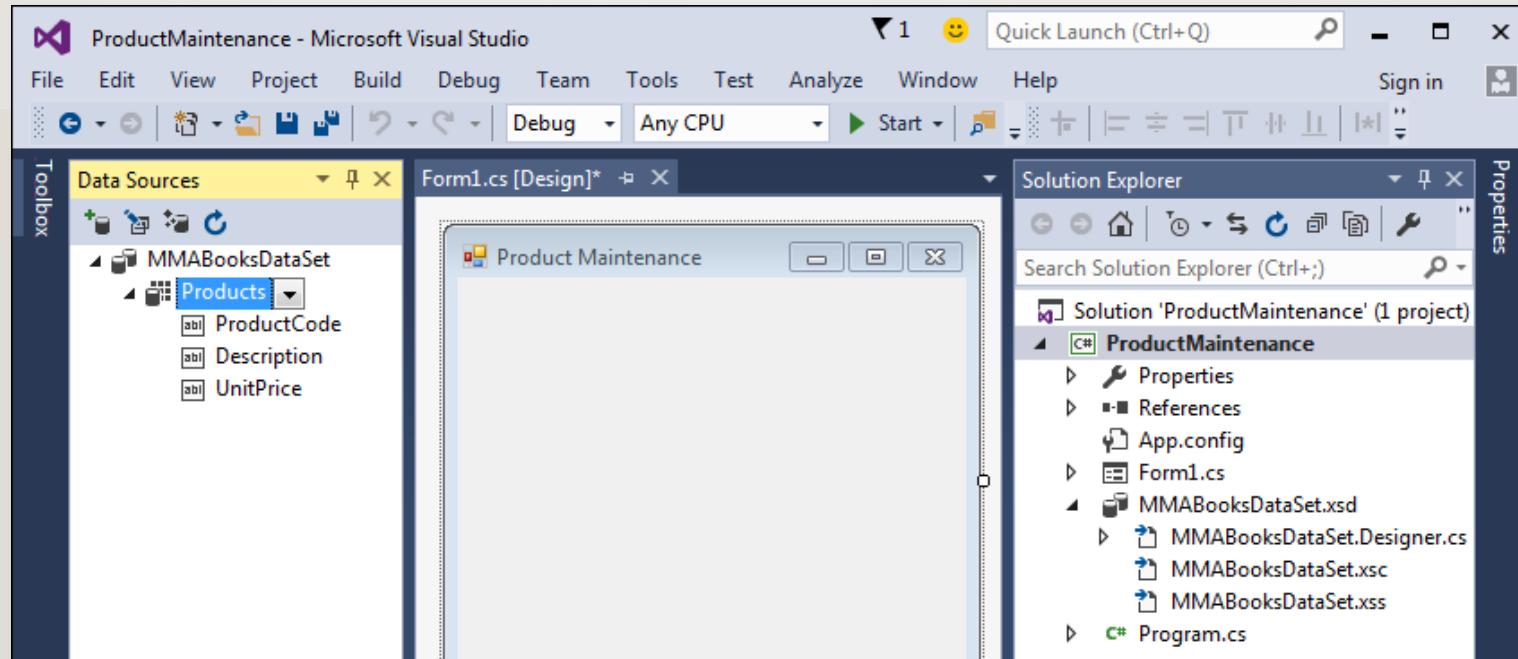
AN EMPTY DATA SOURCES WINDOW

C18, Slide 15

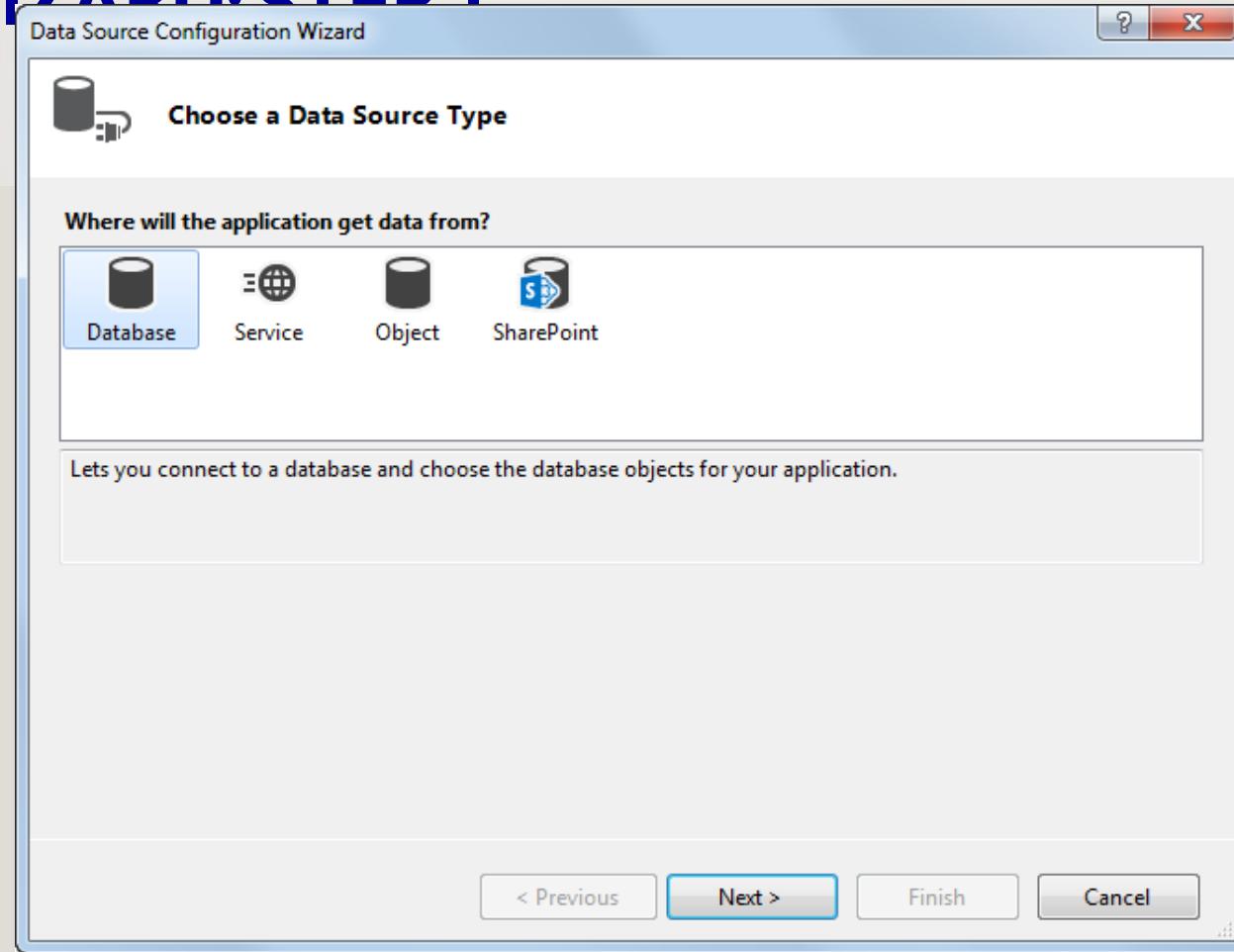


A DATA SOURCES WINDOW AFTER A DATA SOURCE HAS BEEN ADDED

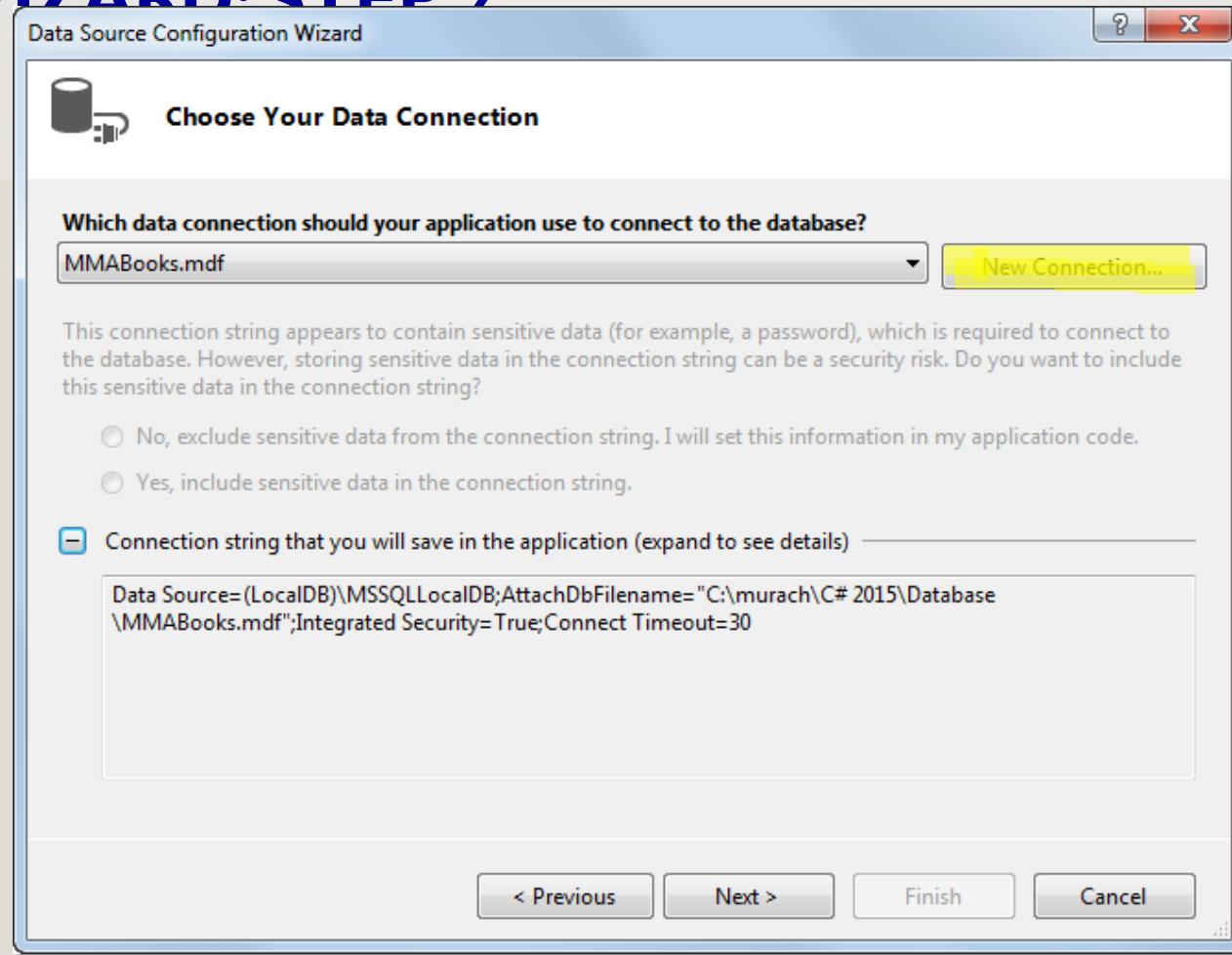
C18, Slide 16



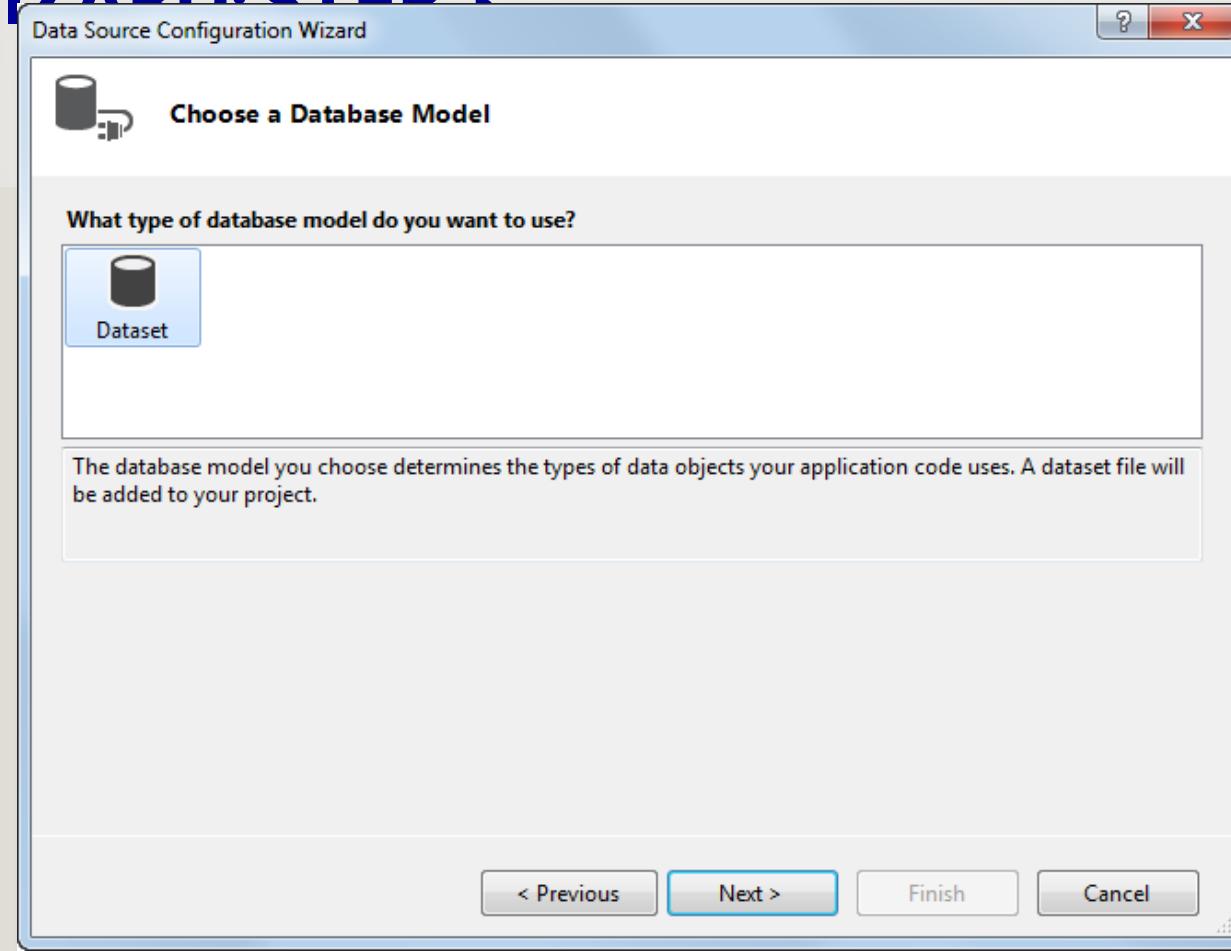
THE DATA SOURCE CONFIGURATION WIZARD. STEP 1



THE DATA SOURCE CONFIGURATION WIZARD·STEP 2

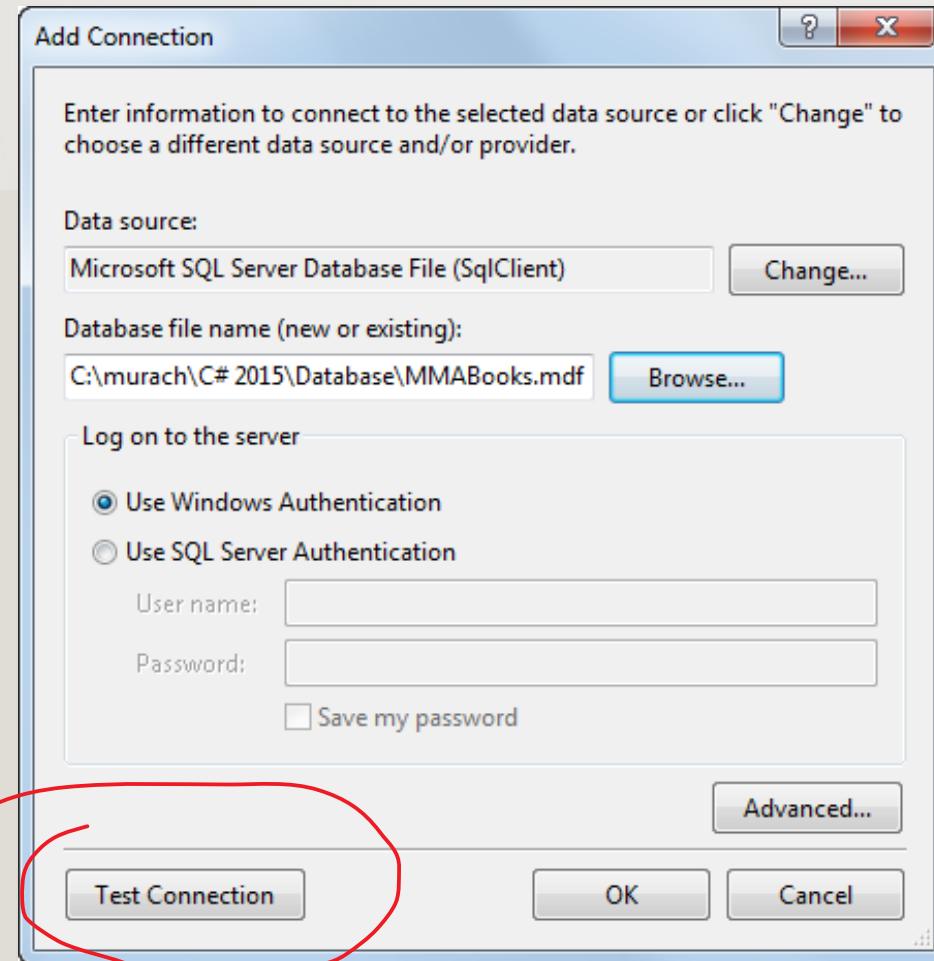


THE DATA SOURCE CONFIGURATION WIZARD, STEP 2



THE ADD CONNECTION DIALOG BOX

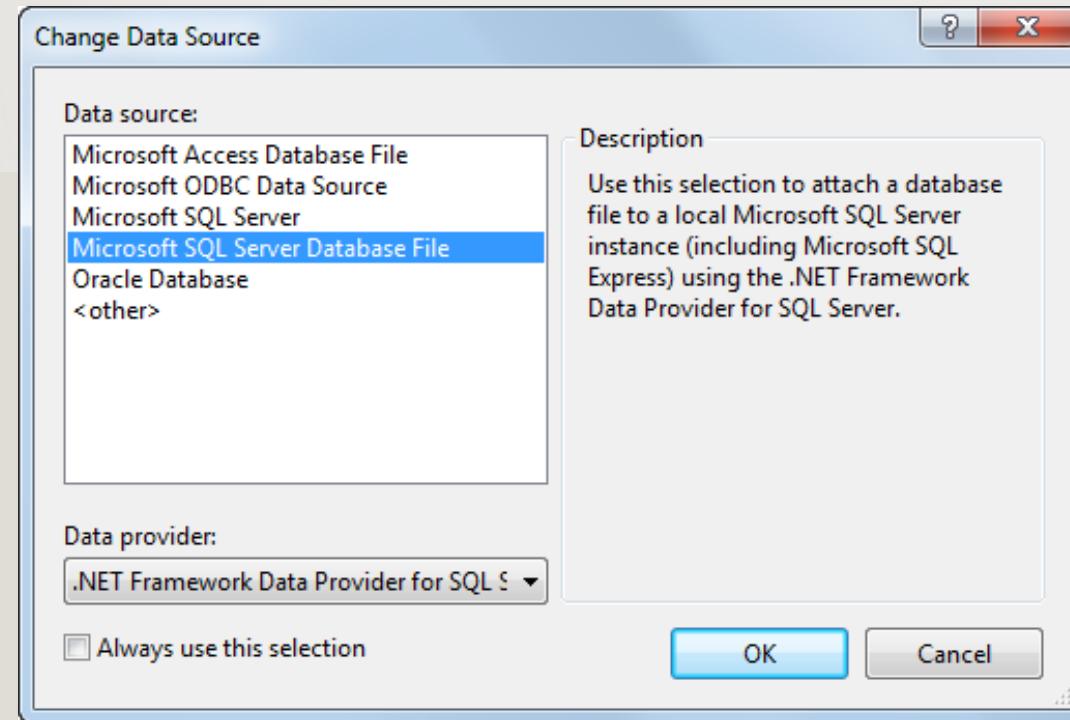
C18, Slide 20



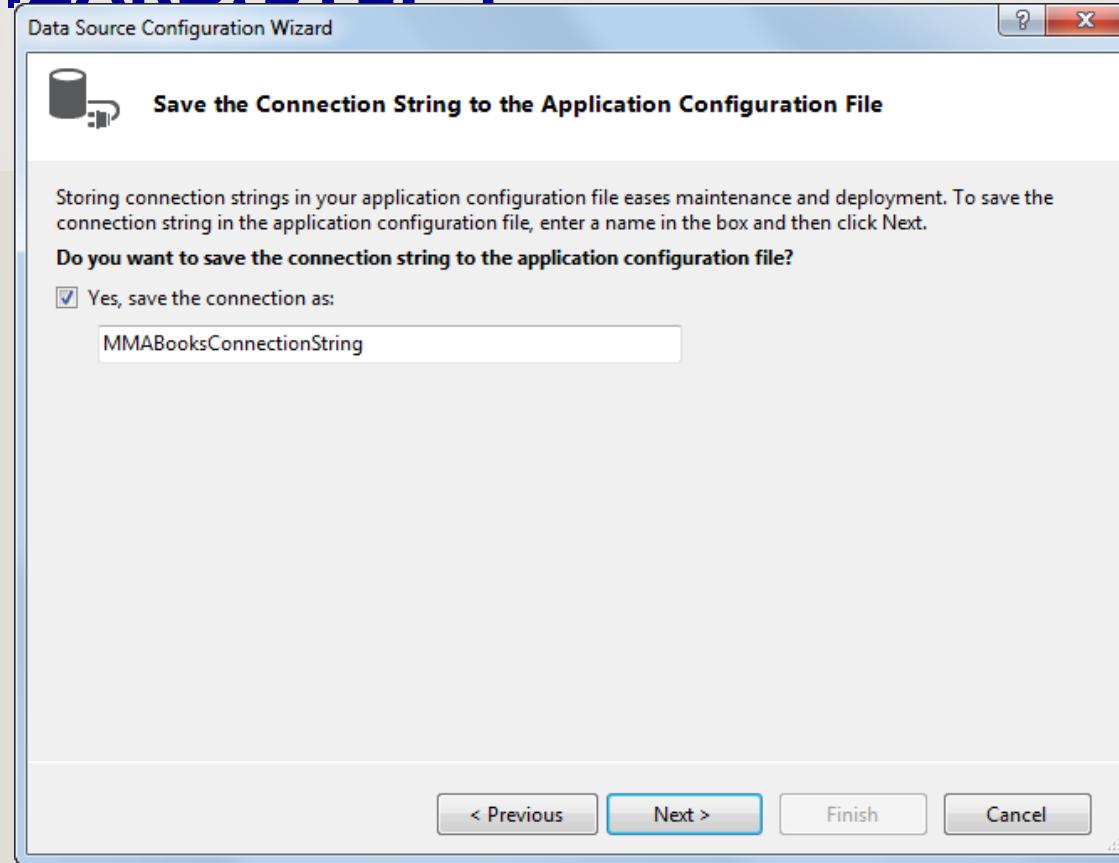
Test firstly in SQL management studio if failed.

THE CHANGE DATA SOURCE DIALOG BOX

C18, Slide 21



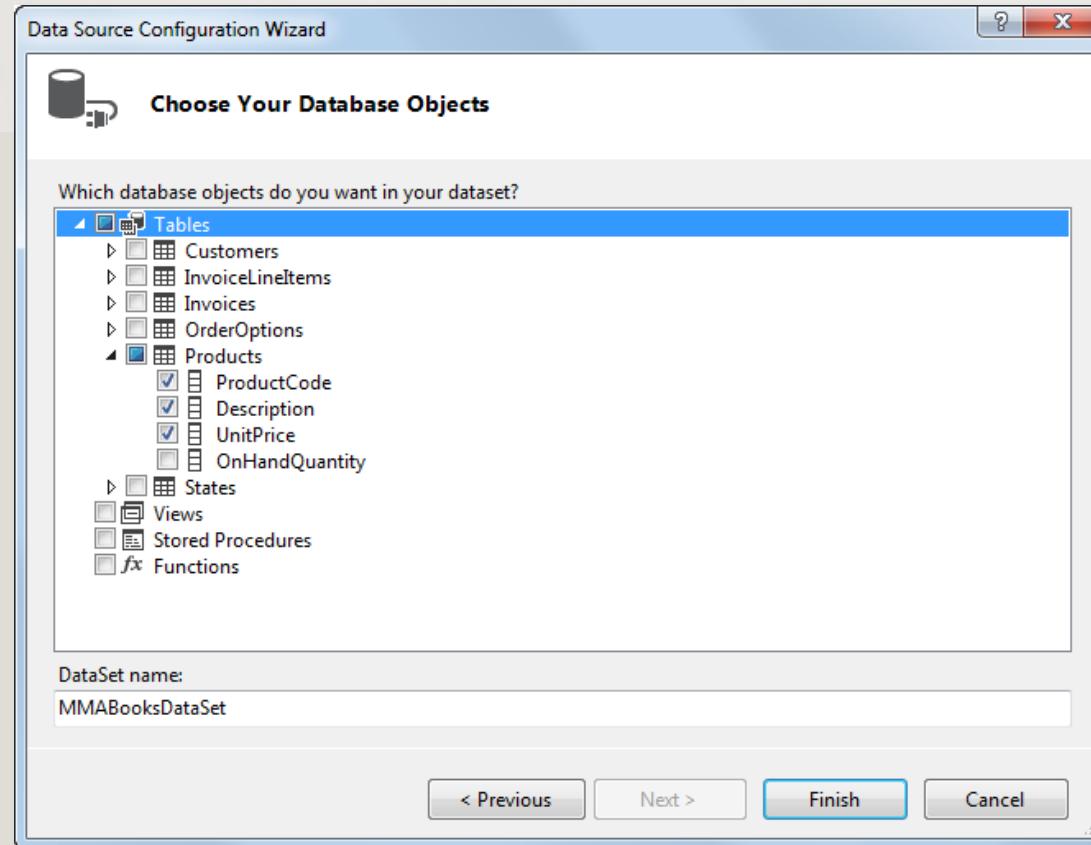
THE DATA SOURCE CONFIGURATION WIZARD: STEP 4



THE INFORMATION THAT'S STORED IN THE APP.CONFIG FILE

```
<connectionStrings>
  <add name="ProductMaintenance.Properties.Settings.MMABooksConnectionString"
    connectionString="Data Source=(LocalDB)\MSSQLLocalDB;
                      AttachDbFilename=|DataDirectory|\MMABooks.mdf;
                      Integrated Security=True;
                      Connect Timeout=30"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

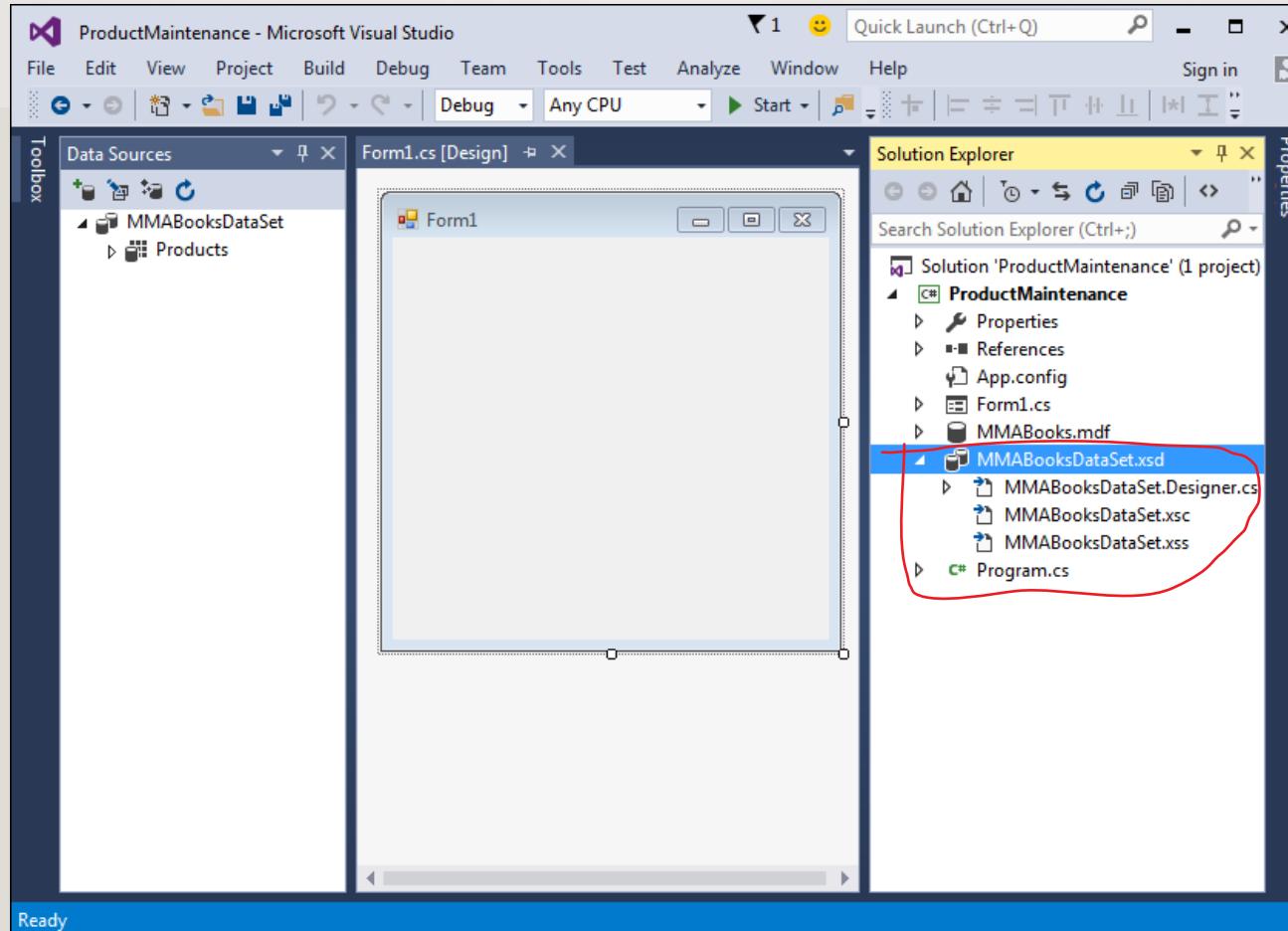
THE DATA SOURCE CONFIGURATION WIZARD: STEP 5



HOW TO WORK WITH COLUMNS THAT HAVE DEFAULT VALUES

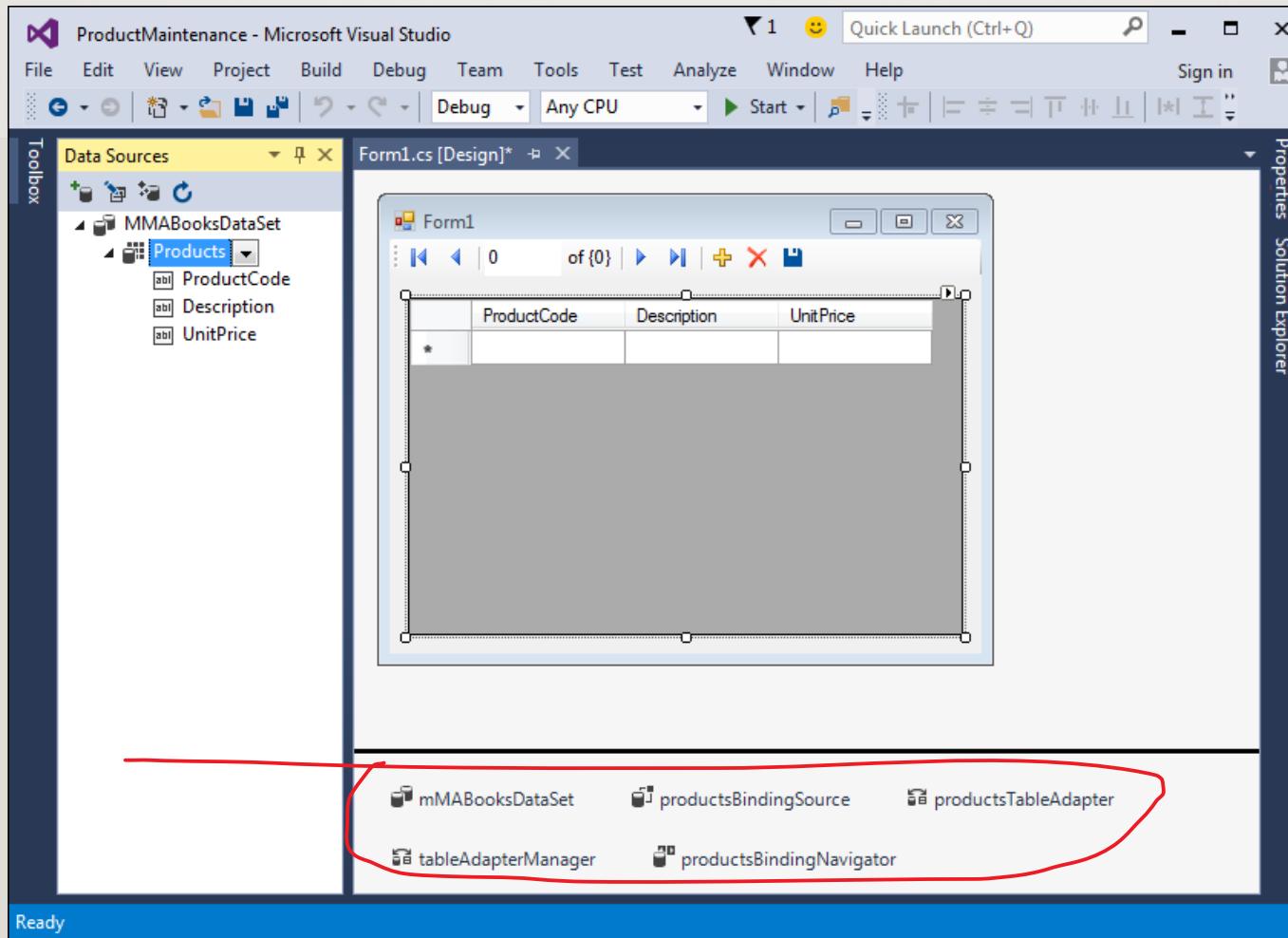
- Omit columns with default values from the dataset unless they're needed by the application.
- Provide values for those columns whenever a row is added to the dataset.

A PROJECT WITH A DATASET DEFINED BY A DATA SOURCE



A FORM WITH THE PRODUCTS TABLE DRAGGED ONTO IT

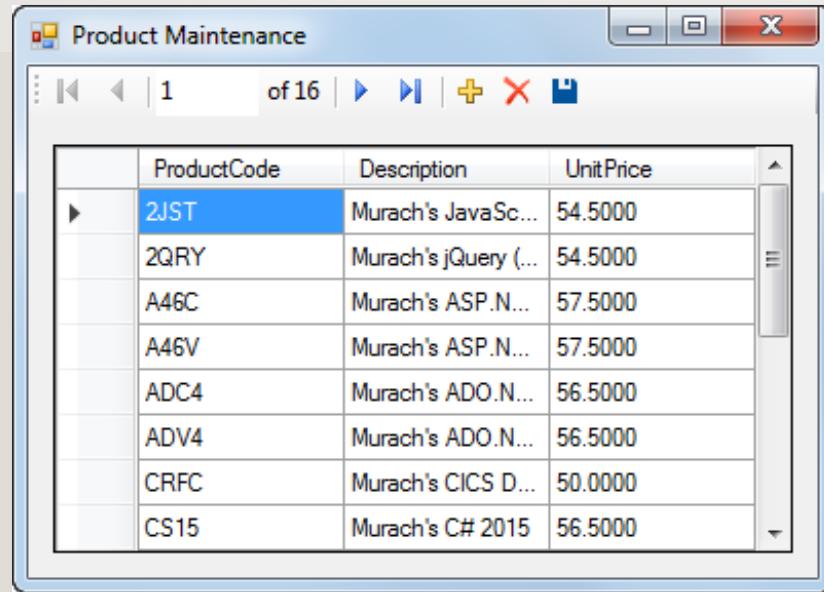
C18, Slide 27



THE CONTROLS AND OBJECTS THAT ARE CREATED WHEN YOU DRAG A DATA SOURCE TO A FORM

- DataGridView control
- BindingNavigator control
- BindingSource object
- DataSet object
- TableAdapter object
- TableAdapterManager object

THE USER INTERFACE FOR THE PRODUCT MAINTENANCE APPLICATION



THE CODE THAT'S GENERATED BY VISUAL STUDIO

C18, Slide 30

```
private void Form1_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the
    // 'mMABooksDataSet.Products' table.
    // You can move, or remove it, as needed.
    this.productsTableAdapter.Fill(
        this.mMABooksDataSet.Products);
}

private void productsBindingNavigatorSaveItem_Click(
    object sender, EventArgs e)
{
    this.Validate();
    this.productsBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(
        this.mMABooksDataSet);
}
```

THE SYNTAX OF THE FILL METHOD

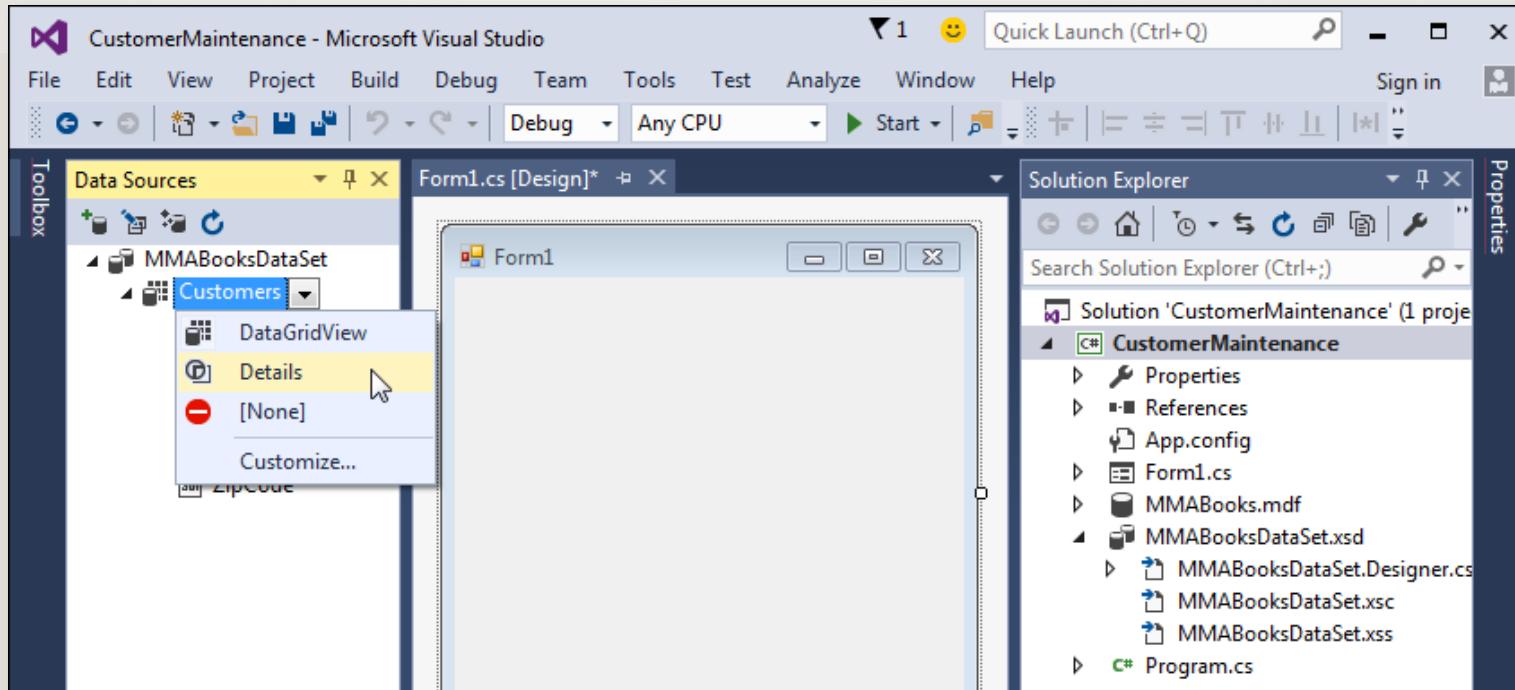
C18, Slide 31

```
tableAdapter.Fill(dataSet.TableName)
```

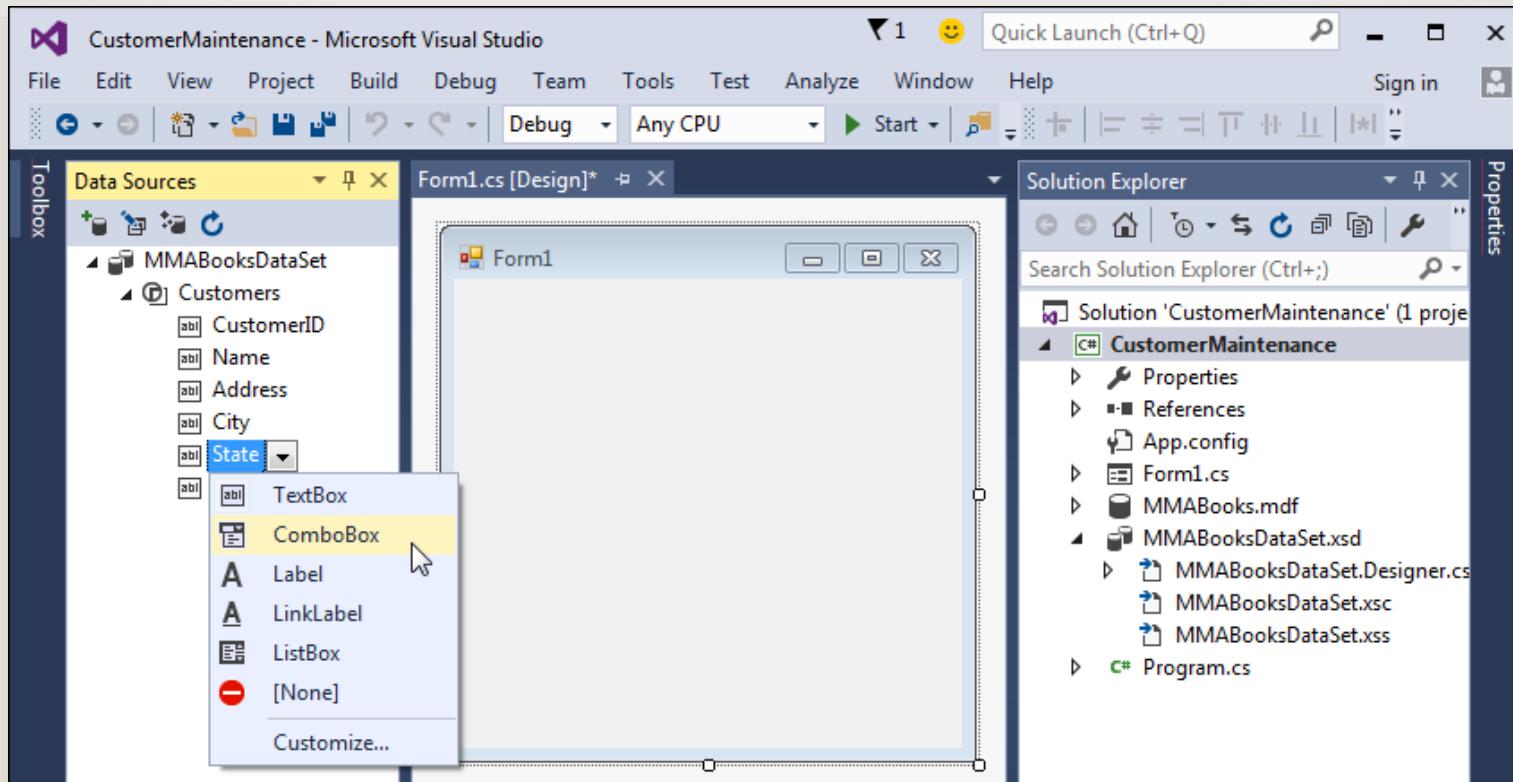
The syntax of the UpdateAll method

```
tableAdapterManager.UpdateAll(dataSet)
```

HOW TO CHANGE THE DEFAULT CONTROL FOR A DATA TABLE

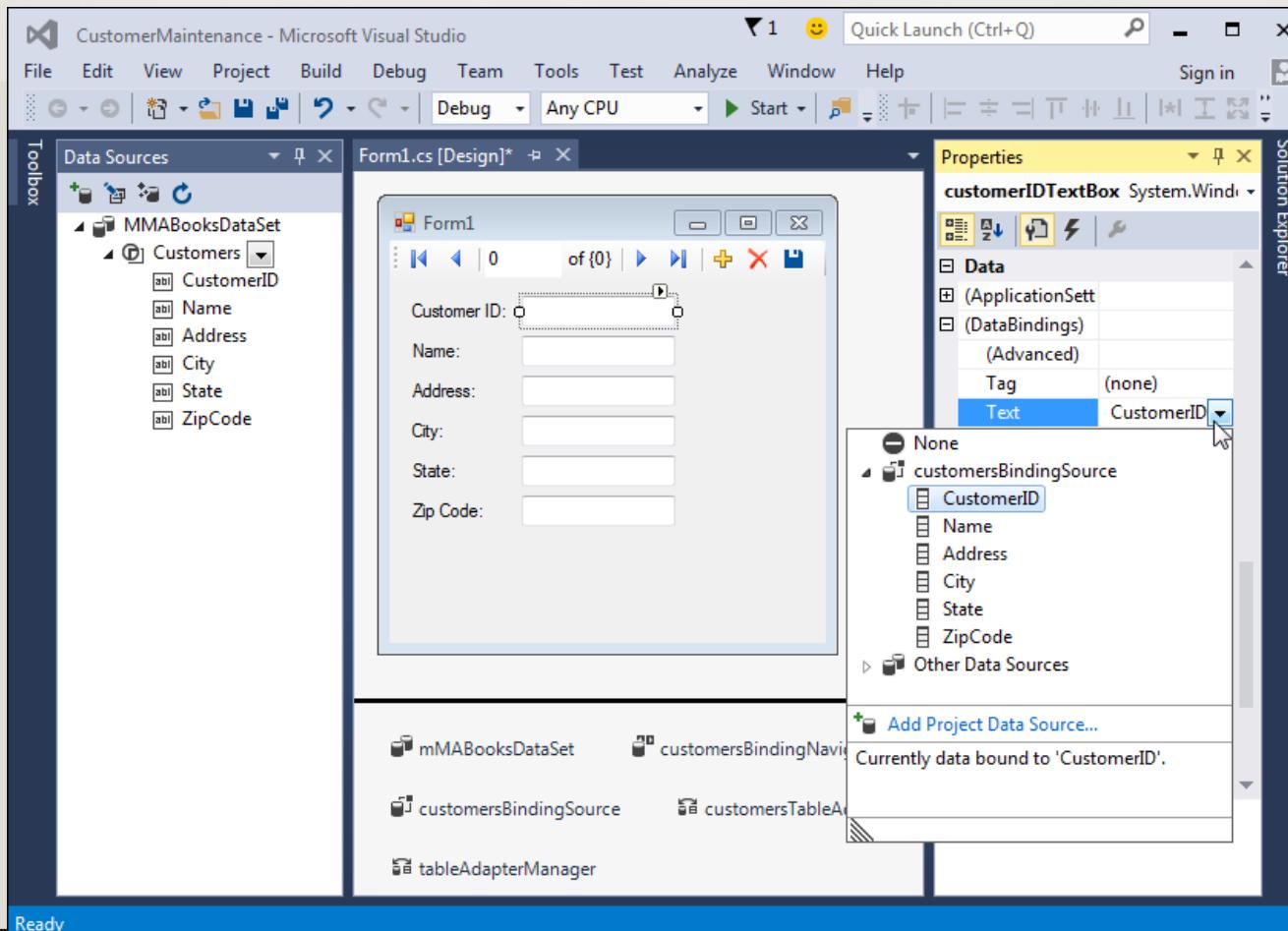


HOW TO CHANGE THE DEFAULT CONTROL FOR A COLUMN IN A DATA TABLE

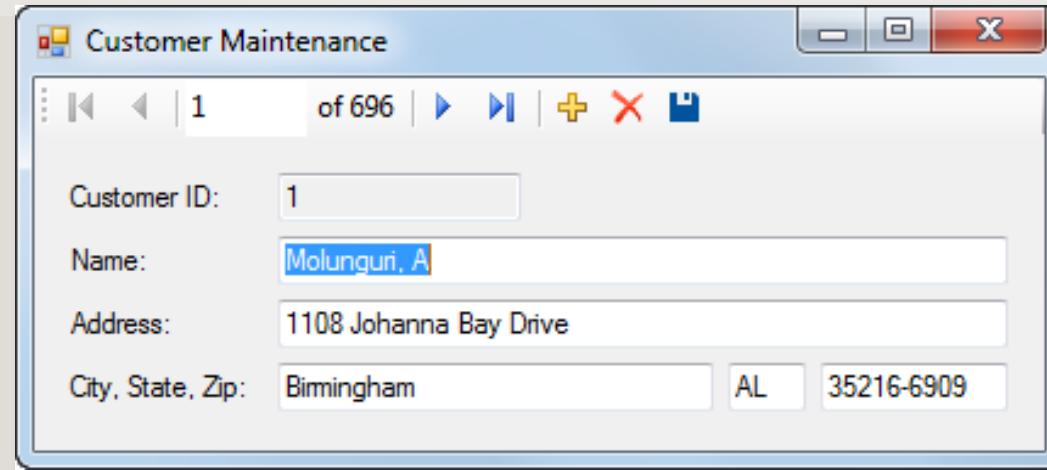


A FORM WITH THE CUSTOMERS TABLE DRAGGED ONTO IT

C18, Slide 34



THE USER INTERFACE FOR THE CUSTOMER MAINTENANCE APPLICATION



THE CODE FOR THE APPLICATION

C18, Slide 36

```
private void Form1_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the
    // 'mMABooksDataSet.Customers' table.
    // You can move, or remove it, as needed.
    this.customersTableAdapter.Fill(
        this.mMABooksDataSet.Customers);
}

private void customersBindingNavigatorSaveItem_Click(
    object sender, EventArgs e)
{
    this.Validate();
    this.customersBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(
        this.mMABooksDataSet);
}
```

.NET DATA PROVIDER EXCEPTION CLASSES

`SqlException`

`OdbcException`

`OleDbException`

COMMON MEMBERS OF THE .NET DATA PROVIDER EXCEPTION CLASSES

Number

Message

Source

Errors

GetType()

CODE THAT CATCHES A SQL EXCEPTION

C18, Slide 39

```
private void Form1_Load(object sender, EventArgs e)
{
    try
    {
        this.customersTableAdapter.Fill(
            this.mMABooksDataSet.Customers);
    }
    catch (SqlException ex)
    {
        MessageBox.Show("Database error # " + ex.Number +
            ": " + ex.Message, ex.GetType().ToString());
    }
}
```

COMMON ADO.NET EXCEPTION CLASSES

C18, Slide 40

`DBConcurrencyException`

`DataException`

`ConstraintException`

`NoNullAllowedException`

Common members of the ADO.NET classes

`Message`

`GetType()`

CODE THAT HANDLES ADO.NET ERRORS

C18, Slide 41

```
try
{
    this.customersBindingSource.EndEdit();
    this.tableAdapterManager.UpdateAll(this.mMABooksDataSet);
}
catch (DBConcurrencyException)
{
    MessageBox.Show("A concurrency error occurred. " +
        "Some rows were not updated.", "Concurrency Exception");
    this.customersTableAdapter.Fill(this.mMABooksDataSet.Customers);
}
catch (DataException ex)
{
    MessageBox.Show(ex.Message, ex.GetType().ToString());
    customersBindingSource.CancelEdit();
}
catch (SqlException ex)
{
    MessageBox.Show("Database error # " + ex.Number +
        ": " + ex.Message, ex.GetType().ToString());
}
```

AN EVENT OF THE DATAGRIDVIEW CONTROL

DataError

**Three properties of the
DataGridViewDataErrorEventArgs class**

Exception

RowIndex

ColumnIndex

CODE THAT HANDLES A DATA ERROR FOR A DATAGRIDVIEW CONTROL

```
private void productsDataGridView_DataError(
    object sender, DataGridViewDataErrorEventArgs e)
{
    int row = e.RowIndex + 1;
    string errorMessage = "A data error occurred.\n" +
        "Row: " + row + "\n" +
        "Error: " + e.Exception.Message;
    MessageBox.Show(errorMessage, "Data Error");
}
```

ConsoleApplicationRoster - Microsoft Visual Studio (Administrator)

File Edit View Project Build Debug Team Tools Test Analyze Window Help

Debug Any CPU WindowsFormsApplication2 Start

Toolbox

Search Toolbox

WebApplicationContact Components

- Pointer
- DataSetCustomer

WindowsFormsApplication2 Components

- Pointer
- CustomerDataSet
- CustomersTableAdapter
- TableAdapterManager

All Windows Forms

- Pointer
- BackgroundWorker
- BindingNavigator
- BindingSource
- Button
- CheckBox
- CheckedListBox
- ComboBox

CustomerDataSet

- Customers

FormDataSource.cs Program.cs FormDataSource.cs [Design]

FormDataSource

| | CustomerId | CustomerName | CustomerEmail | TicketCredits |
|---|------------|--------------|---------------|---------------|
| * | | | | |

Customer Id:

Customer Name:

Customer Email:

Ticket Credits:

customerDataSet customersBindingSource customersTableAdapter

tableAdapterManager customersBindingNavigator

Output

Error List Output

Solution Explorer

Search Solution Explorer (Ctrl+;)

- Site.Mobile.Master
- ViewSwitcher.ascx
- Web.config
- WebFormDataSource.aspx
- WindowsFormsApplication1
- WindowsFormsApplication2
- Properties
- References
- App.config

Properties

customersDataGridView System.Windows.Forms.DataGridView

CursorChanged

DataBindingComplete

DataError

DataMemberChanged

DataSourceChanged

DefaultCellStyleChanged

DefaultValuesNeeded

DockChanged

Edit Columns... Add Column...

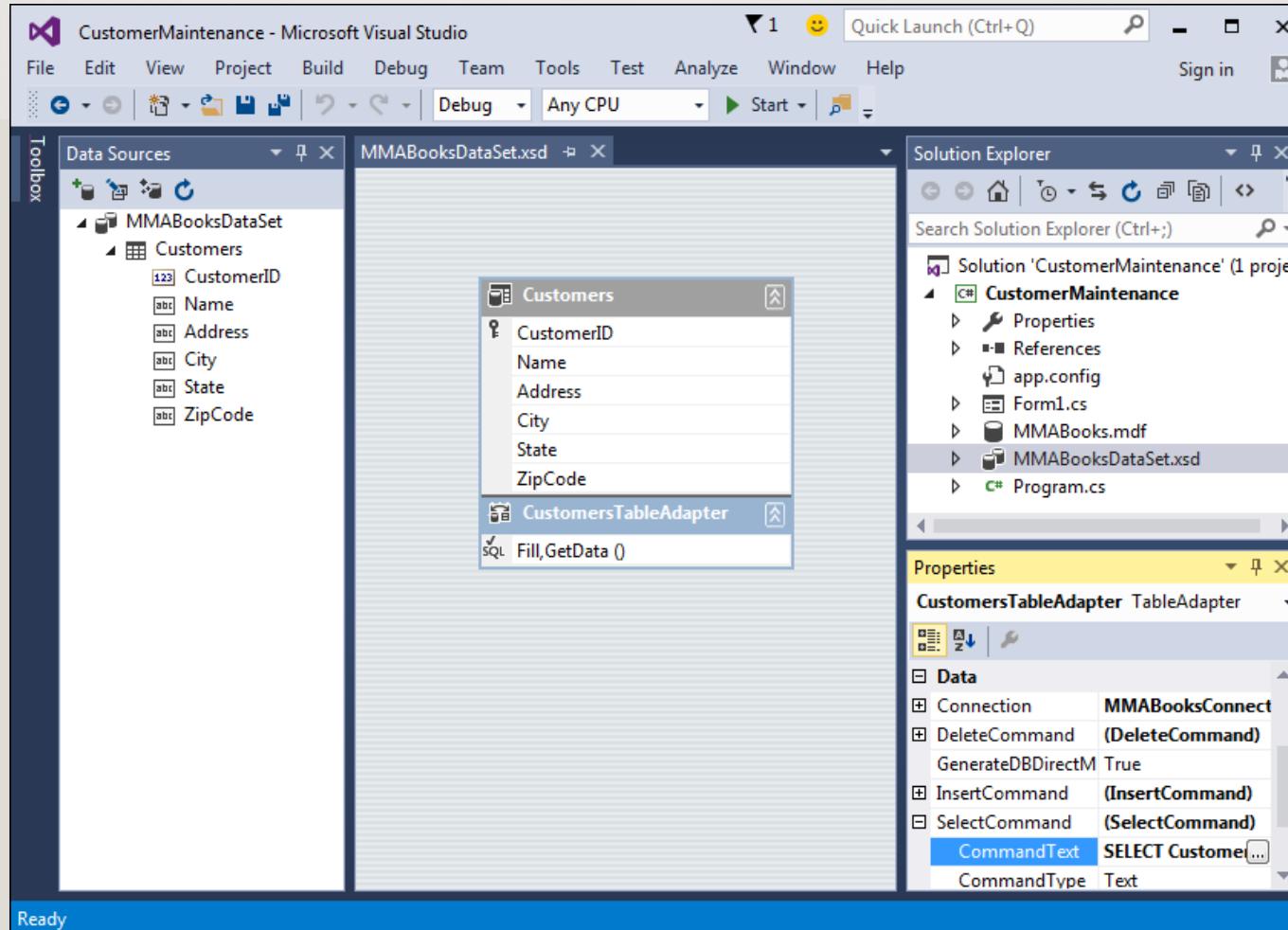
DataError

Occurs when an external data-parsing or validation operation throws an exception, or when an attempt to commit data to a data source does not succeed...

Add to Source Control

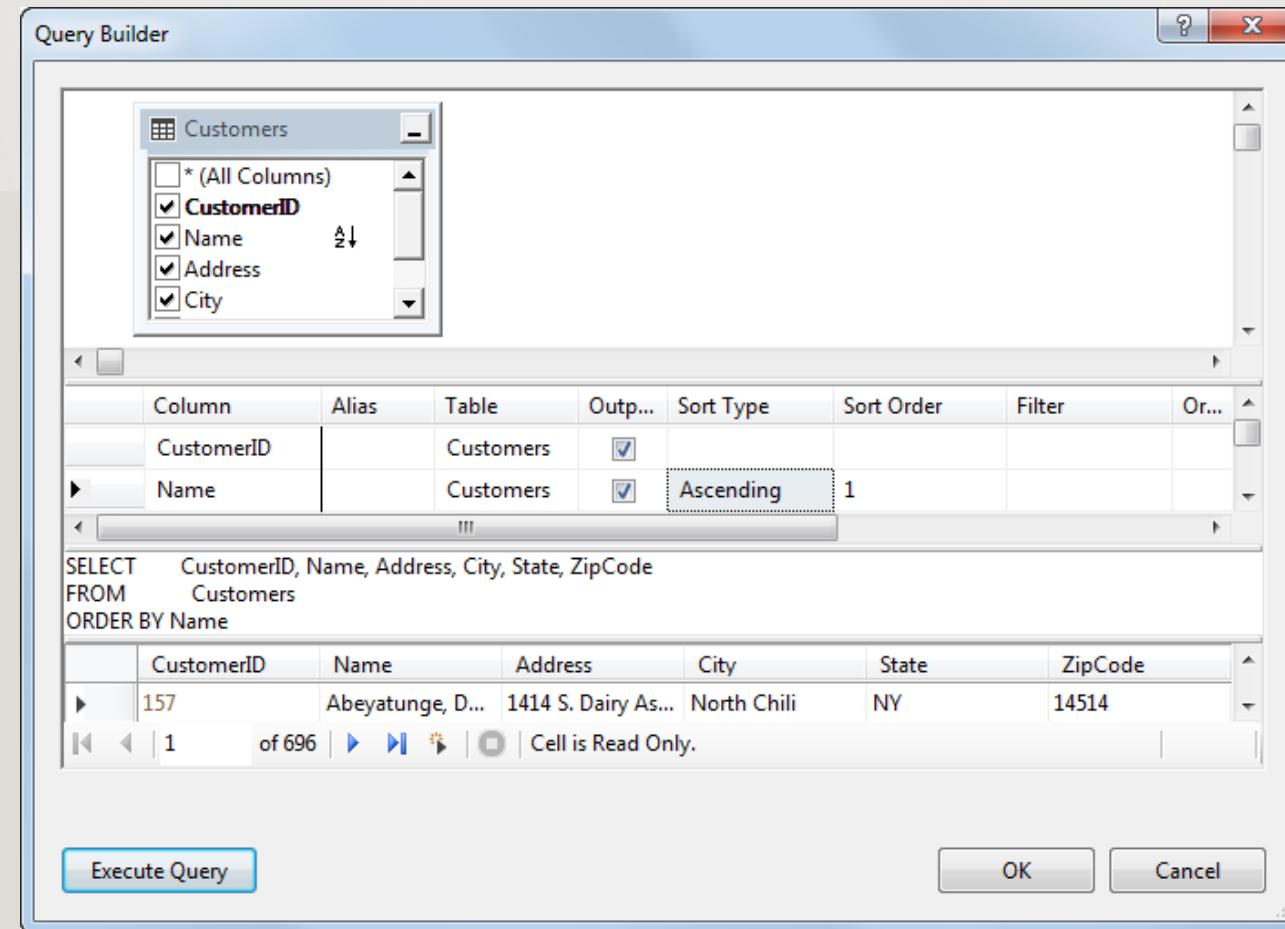
THE SCHEMA DISPLAYED IN THE DATASET DESIGNER

C18, Slide 45



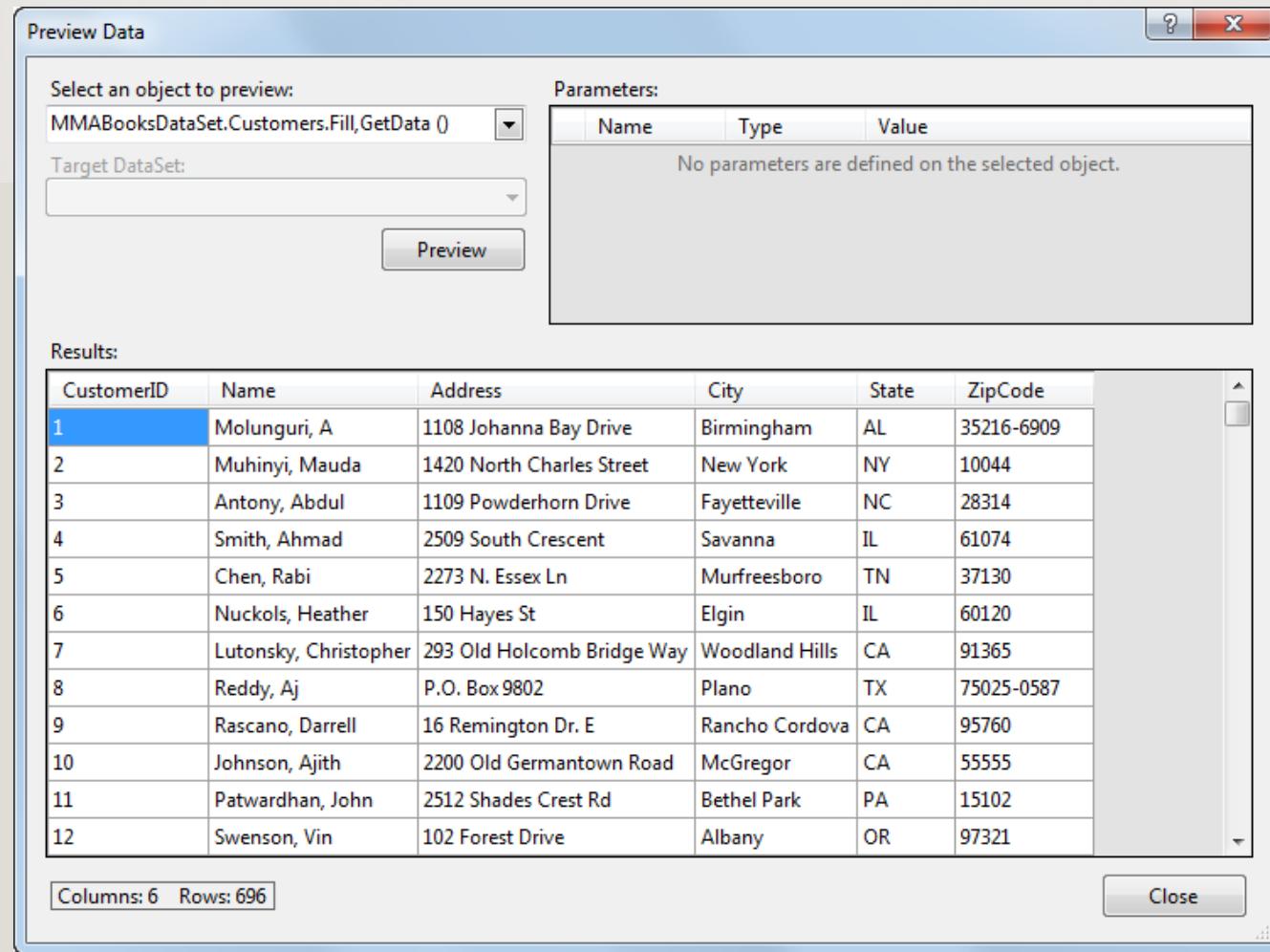
THE QUERY BUILDER

C18, Slide 46

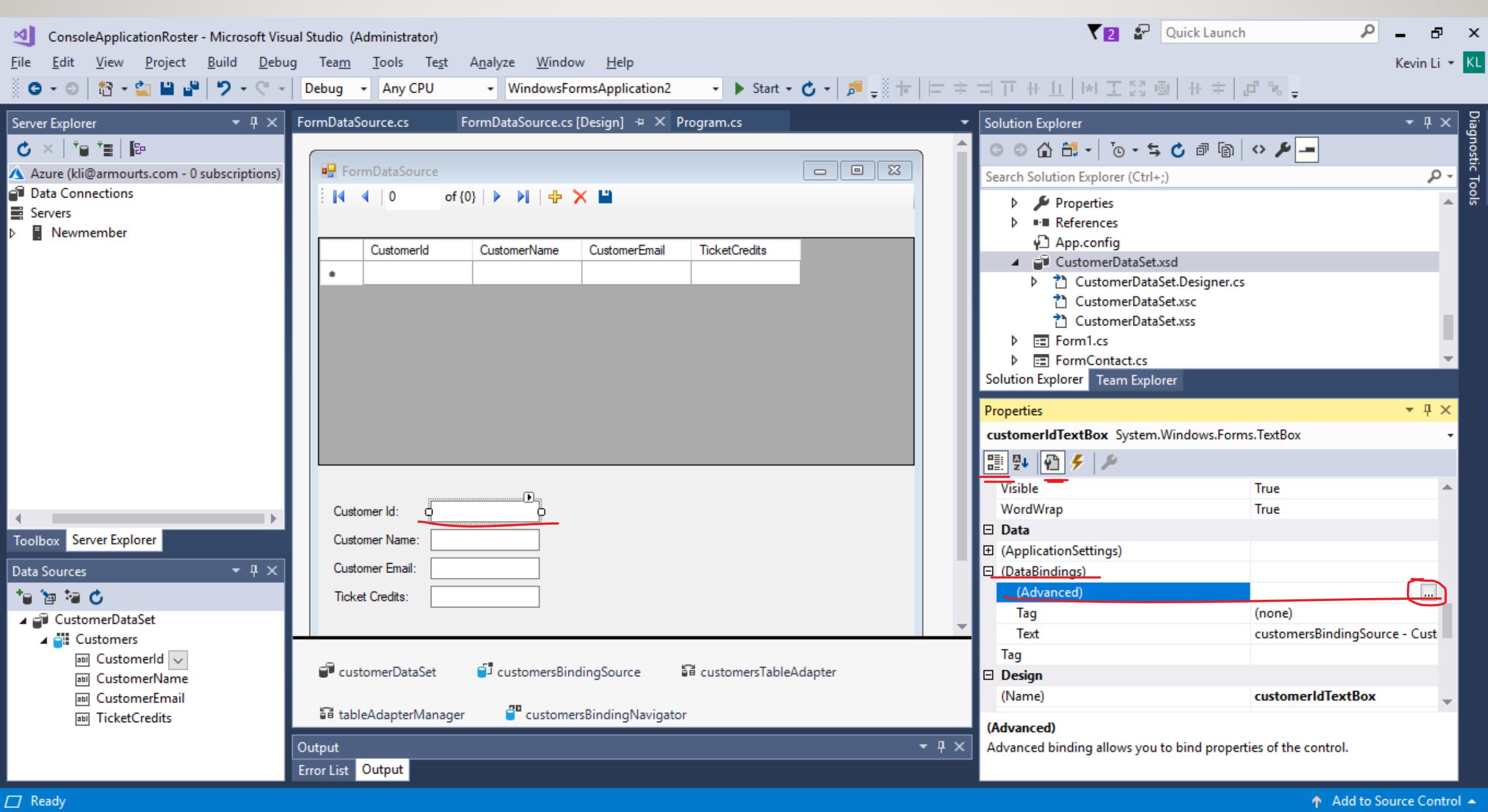


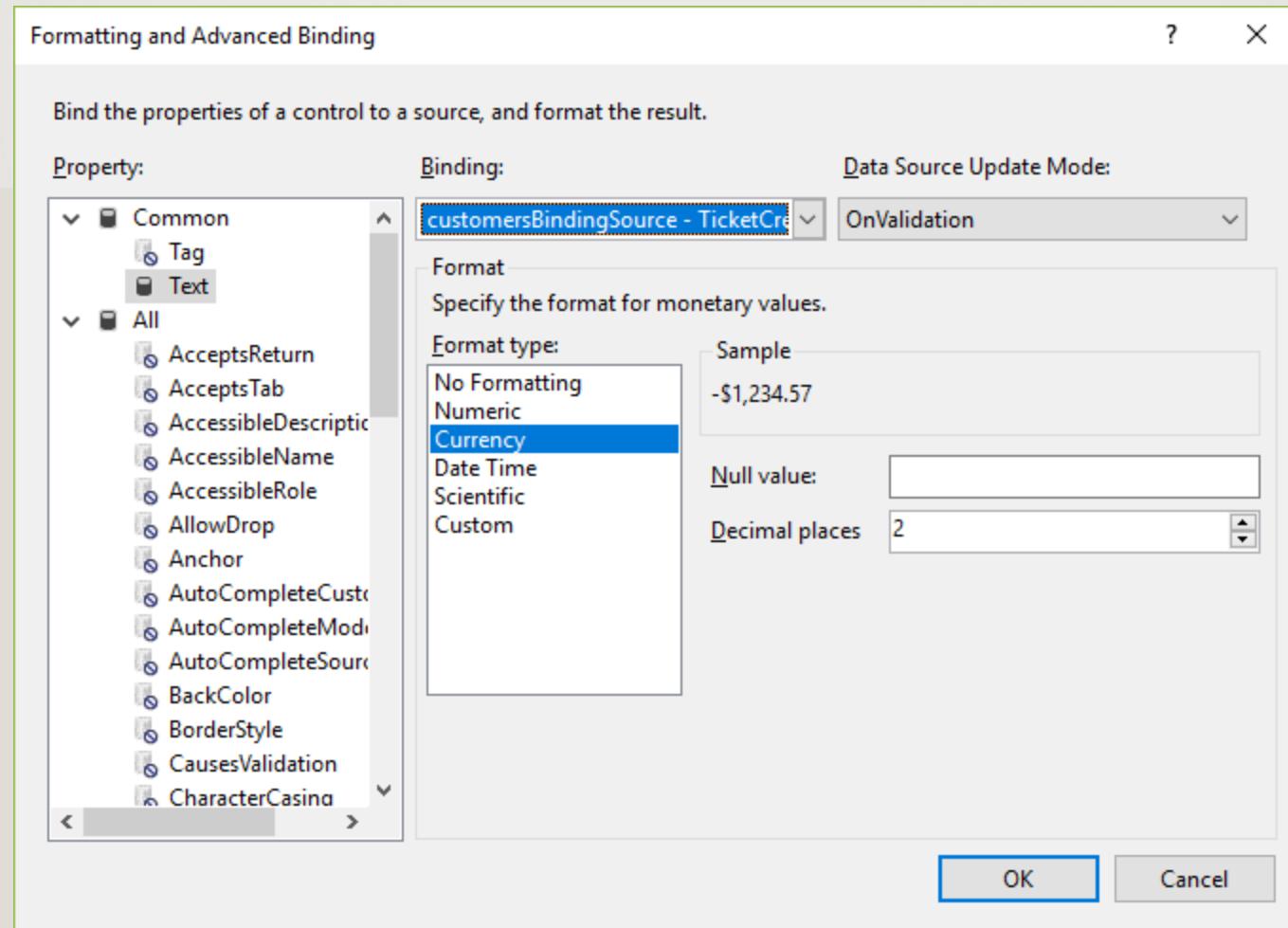
THE PREVIEW DATA DIALOG BOX

C18, Slide 47



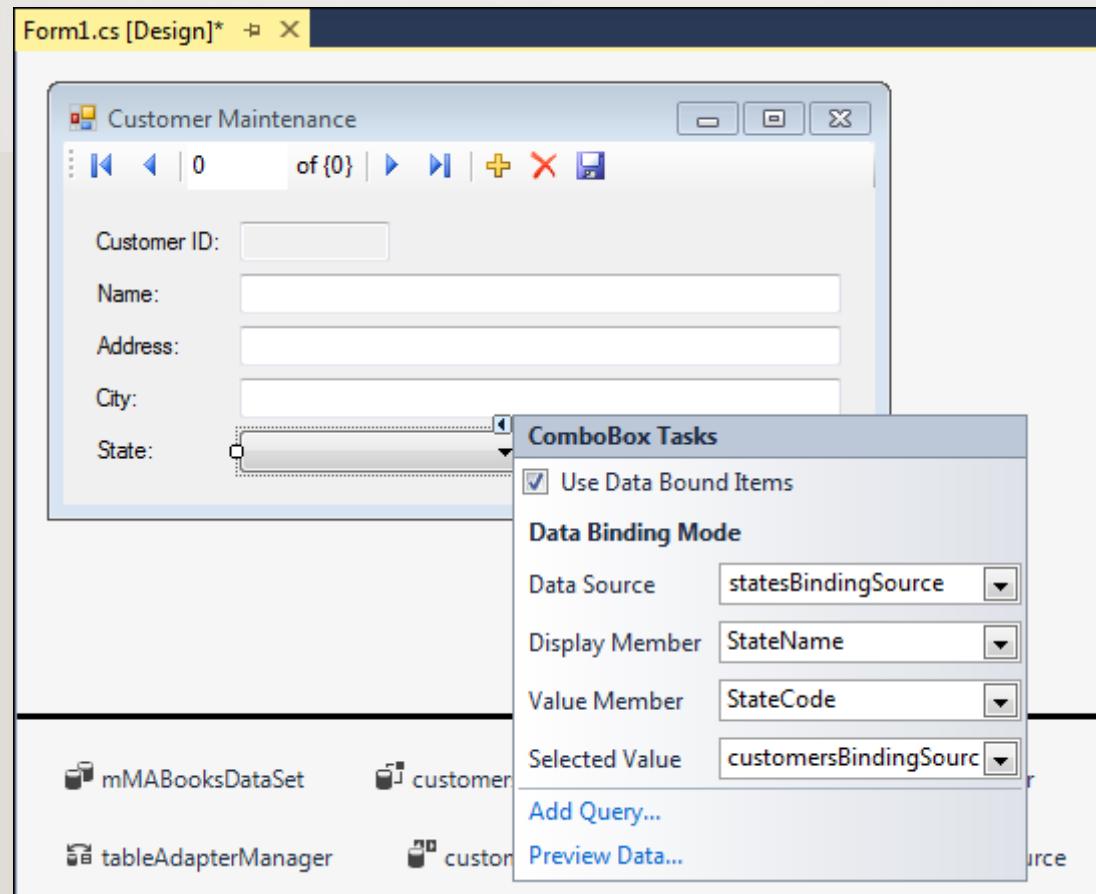
Learn the data manipulation steps from the generated SQL queries.





A COMBO BOX THAT'S BOUND TO A DATA SOURCE

C19, Slide 51



COMBO BOX PROPERTIES FOR BINDING

C19, Slide 52

DataSource

DisplayMember

ValueMember

SelectedValue

COMMON PROPERTIES OF THE BINDINGSOURCE CLASS

Position

Count

COMMON METHODS OF THE BINDINGSOURCE CLASS

- AddNew()**
- EndEdit()**
- CancelEdit()**
- RemoveCurrent()**
- MoveFirst()**
- MovePrevious()**
- MoveNext()**
- MoveLast()**

A STATEMENT THAT ADDS A NEW ROW TO A DATA SOURCE

```
this.customersBindingSource.AddNew();
```

A statement that saves the changes to the current row and ends the edit

```
this.customersBindingSource.EndEdit();
```

A statement that cancels the changes to the current row

```
this.customersBindingSource.CancelEdit();
```

A statement that removes the current row from a data source

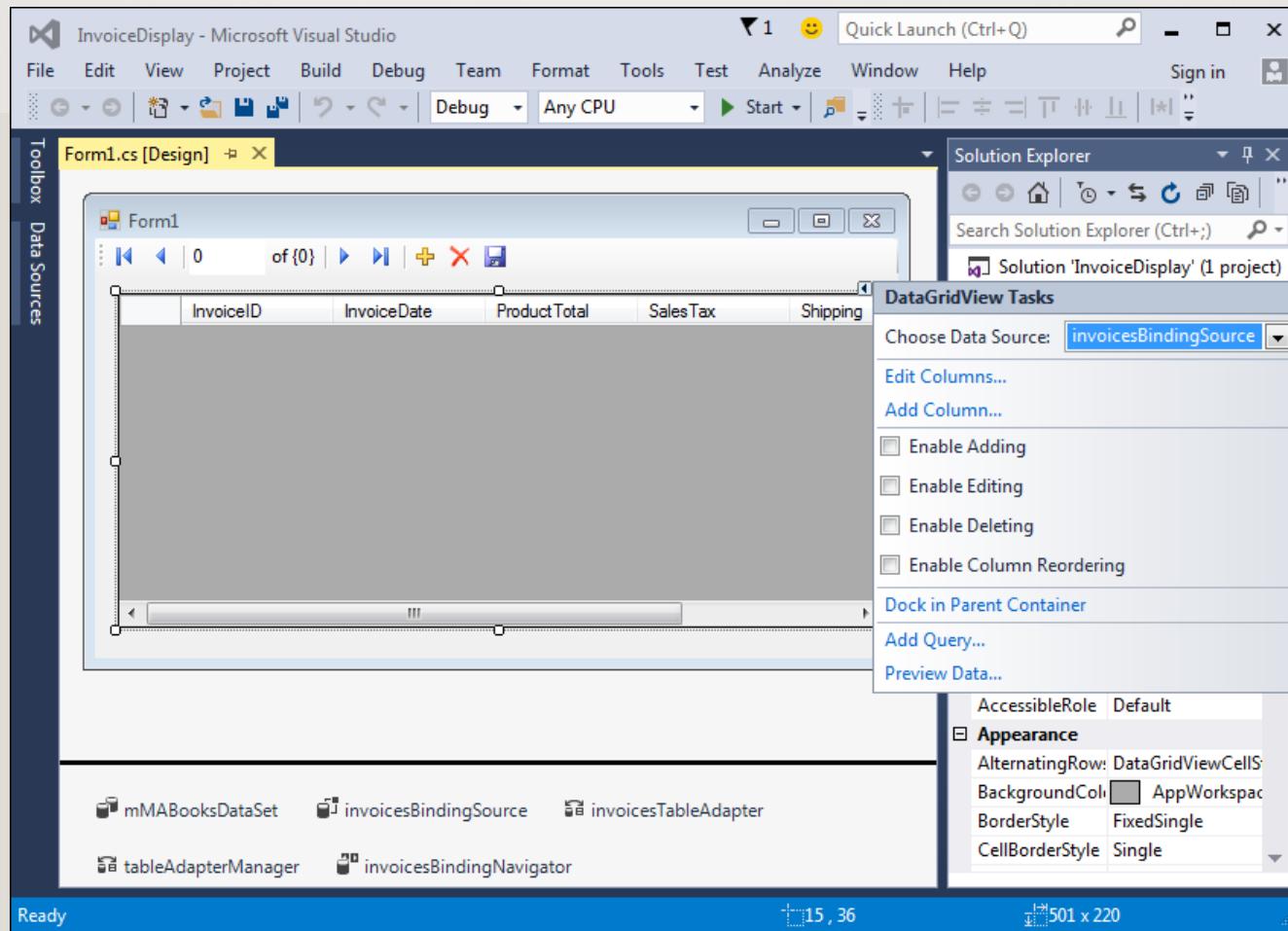
```
this.customersBindingSource.RemoveCurrent();
```

CODE THAT MOVES TO THE NEXT ROW AND DISPLAYS THE POSITION AND COUNT

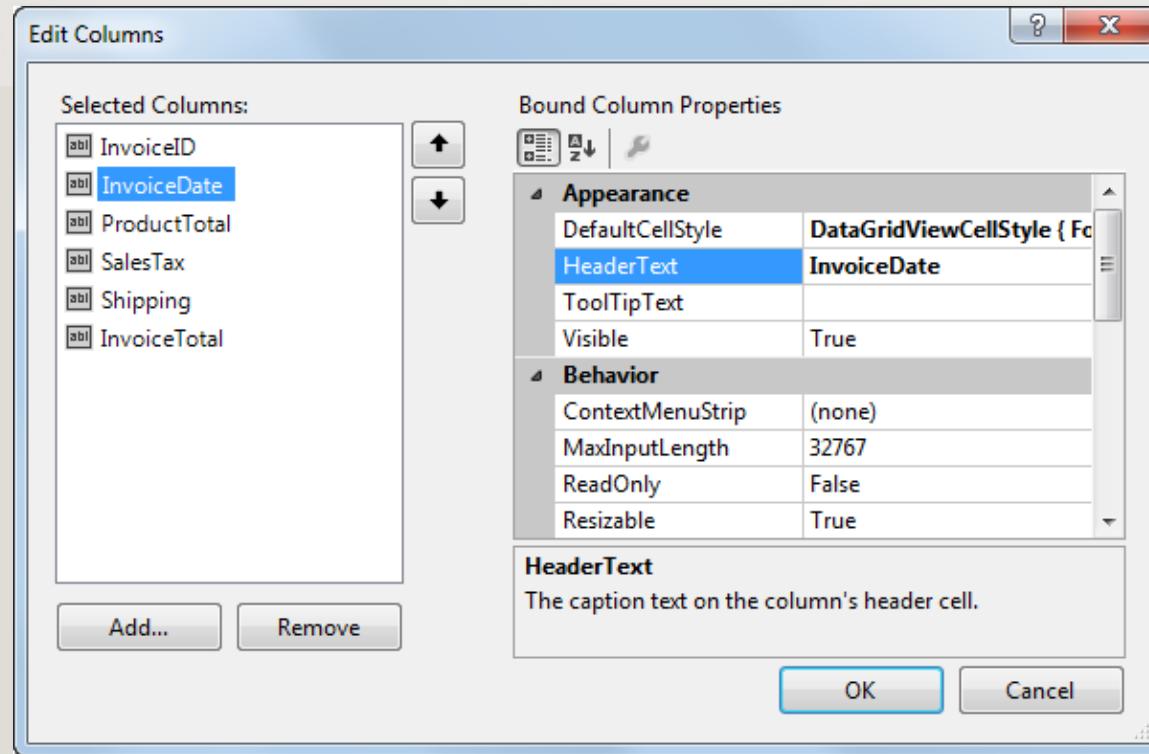
```
private void btnNext_Click(object sender, EventArgs e)
{
    this.customersBindingSource.MoveNext();
    int position = customersBindingSource.Position + 1;
    txtPosition.Text = position + " of " +
        customersBindingSource.Count;
}
```

THE SMART TAG MENU FOR A DATAGRIDVIEW CONTROL

C19, Slide 57



THE DIALOG BOX FOR EDITING DATAGRIDVIEW COLUMNS



COMMON PROPERTIES OF A COLUMN

C19, Slide 59

HeaderText

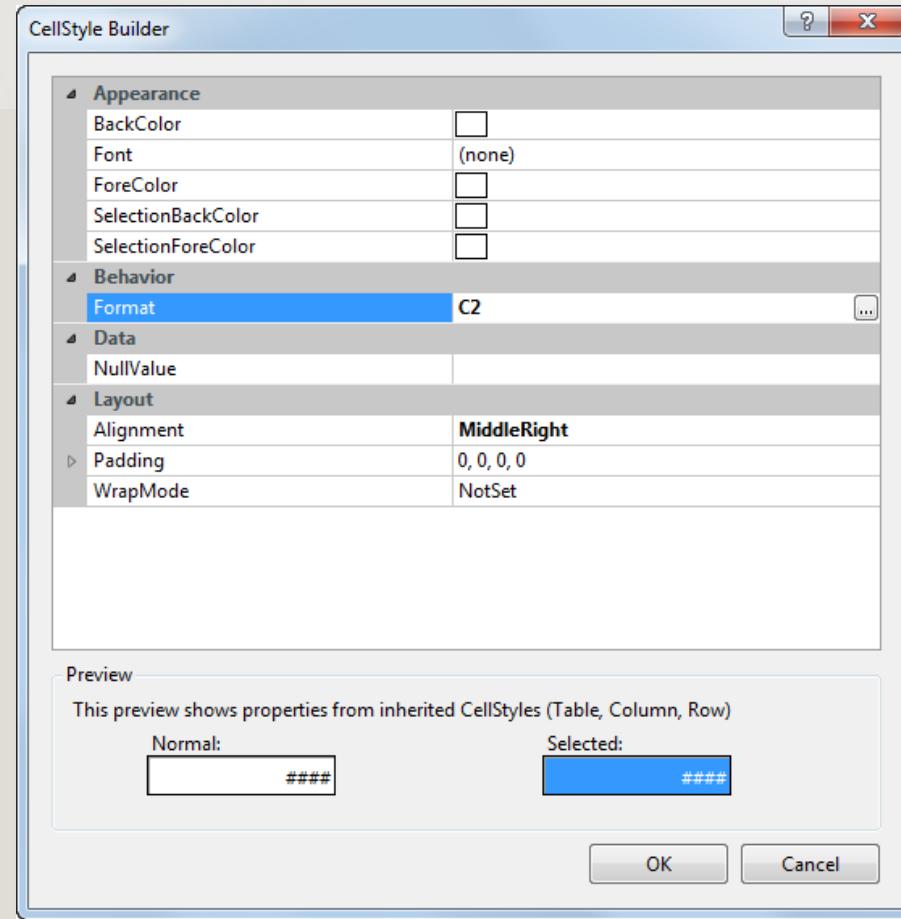
Width

DefaultCellStyle

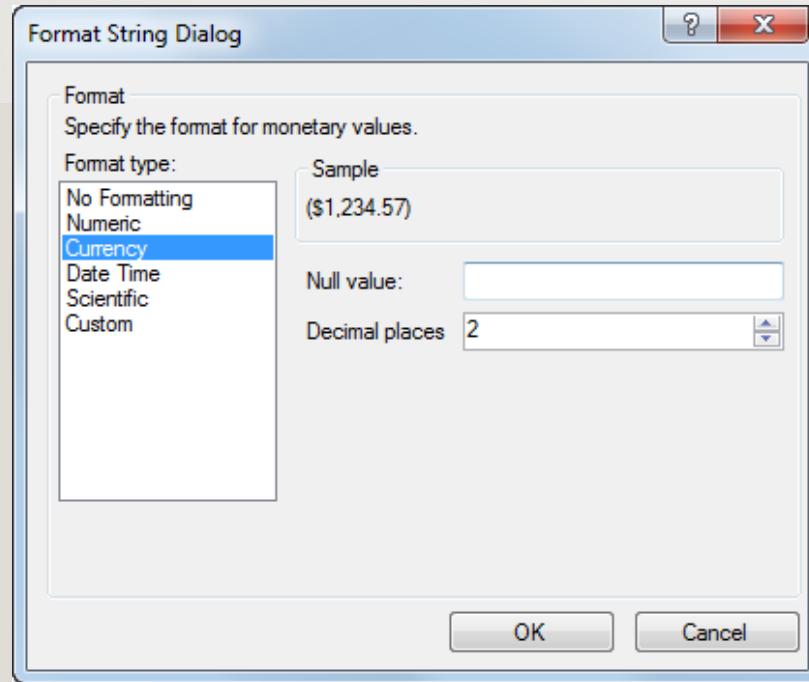
ReadOnly

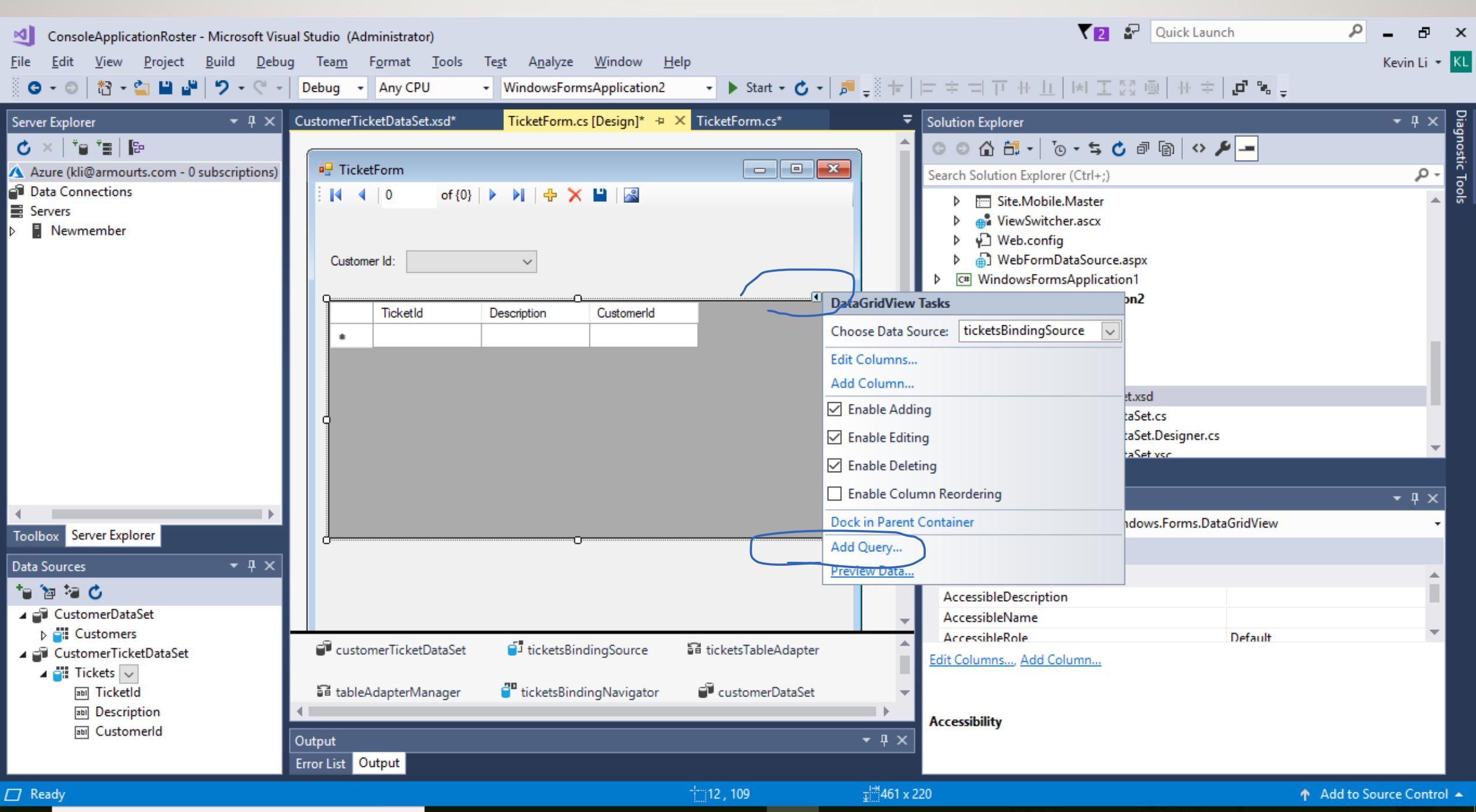
SortMode

TO FORMAT COLUMNS: THE CELLSTYLE BUILDER DIALOG BOX

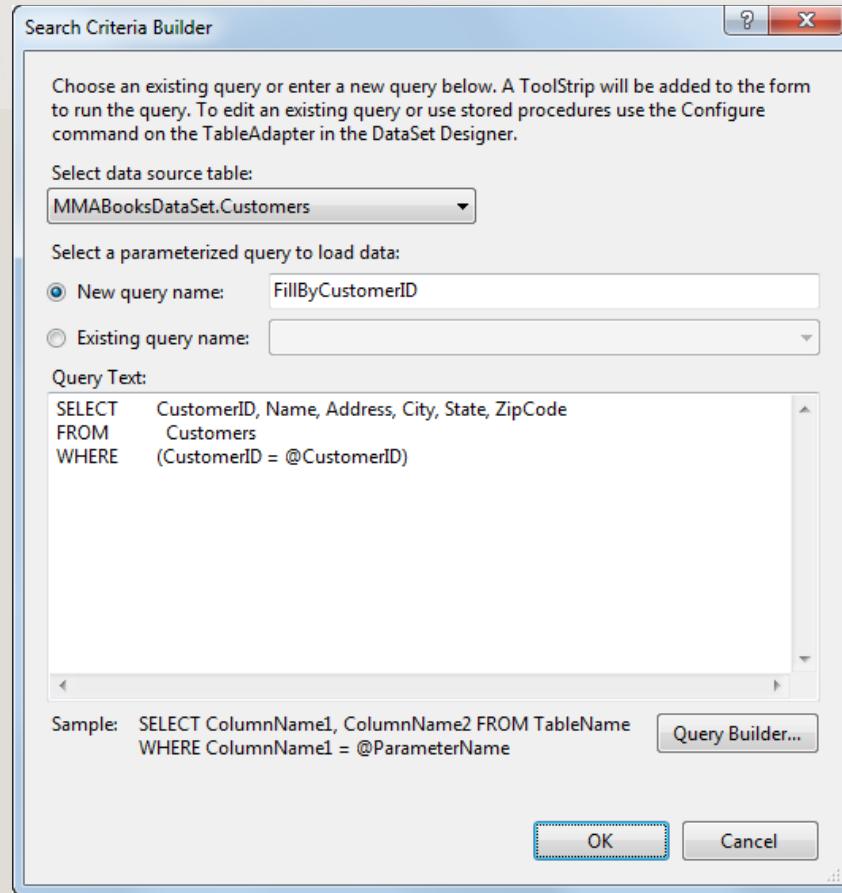


TO FORMAT COLUMNS: THE FORMAT STRING DIALOG BOX





THE DIALOG BOX FOR CREATING A PARAMETERIZED QUERY



DEMO OF ADO.NET IN A MVC/ASP.NET PROJECT

LAST ASSIGNMENT - ADO .NET

Create an app to do CRUD on your SQL database.

Basic requirements:

Create a table named with your name in your SQL database

Do at least one of the CRUD ([Create, read, update and delete](#)) to the SQL database.

Use the basic ADO. NET classes: [SqlConnection](#), [SqlCommand](#), [SqlDataReader](#) and/or [SqlDataAdapter](#) etc.

Please zip and upload your sources code files files(at least the one doing db connection.)

Options for higher scores:

All the CRUD

Use any of these:

Dataset classes

[SqlCommand.Parameters](#)

A separated class doing db connections and the CRUD commands

A data model class to hold your data