Database Management

1. An Introduction to Databases
2. LINQ and Databases

An Introduction to Databases

- Data Source Configuration Wizard
- Accessing a Database Table
- Binding to Additional Tables
- Browsing a Connected Database
- Querying a Table with LINQ
- Primary and Foreign Keys
- The Join of Two Tables

An Introduction to Databases

- Hierarchical databases
- Relational databases
- Object-oriented databases
- Graph databases
- NoSQL databases
- …

Sample Table – Cities Table

name	country	pop2010	pop2015
Bombay	India	20.1	22
Buenos Aires	Argentina	13.1	13.4
Calcutta	India	15.6	17
Delhi	India	17	18.7
Dhaka	Bangladesh	14.8	17
Mexico City	Mexico	19.5	20.2
New York	USA	19.4	20
Sao Paulo	Brazil	19.6	20.1
Shanghai	China	15.8	17.2
Tokyo	Japan	36.1	36.4

Sample Table – Countries Table

name	pop2010	monetaryUnit
Argentina	41.9	peso
Bangladesh	152.6	raka
Brazil	195.2	real
China	1379.7	yuan
India	1196.8	rupee
Indonesia	258.5	rupiah
Japan	129	yen
Mexico	117.4	peso
Pakistan	184.2	rupee
USA	299	dollar

Database Terminology

- A **table** is a rectangular array of data.
- Each column of the table, called a **field**, contains the same type of information.
- Each row, called a **record**, contains all the information about one entry in the table.

Database Management Software (DBMS)

- Used to create databases
- Databases contain one or more related tables
- Examples of DBMS are Access, Oracle, and SQL Server.
- The databases used in this course are created with Access and have the extension *accdb*.

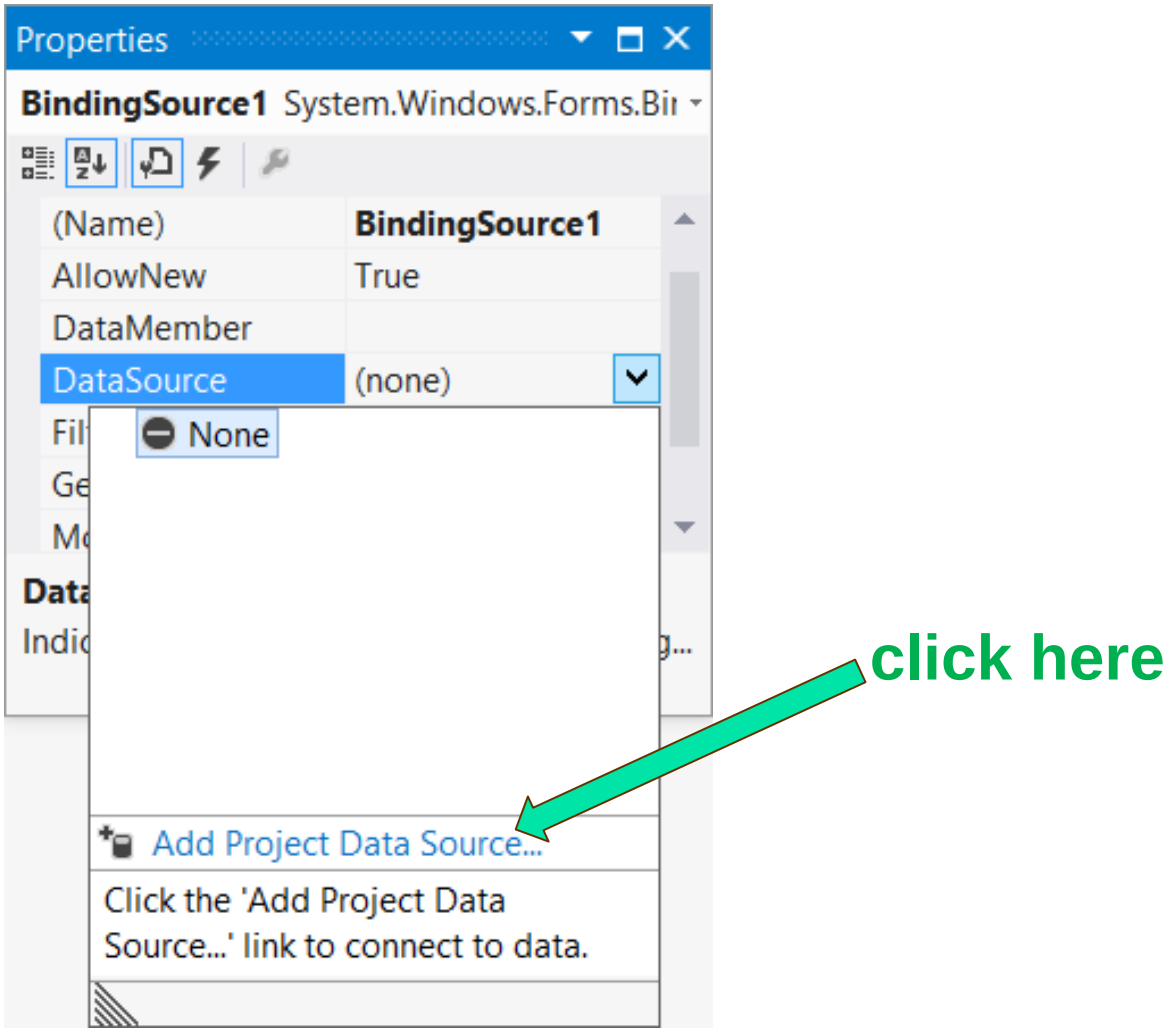
Megacities.accdb

- Contains the two tables Cities and Countries shown earlier.
- This database will be used extensively.
- Several steps are required to bind to a table of the database. We will use the Data Source Configuration Wizard. (See the next slides.)

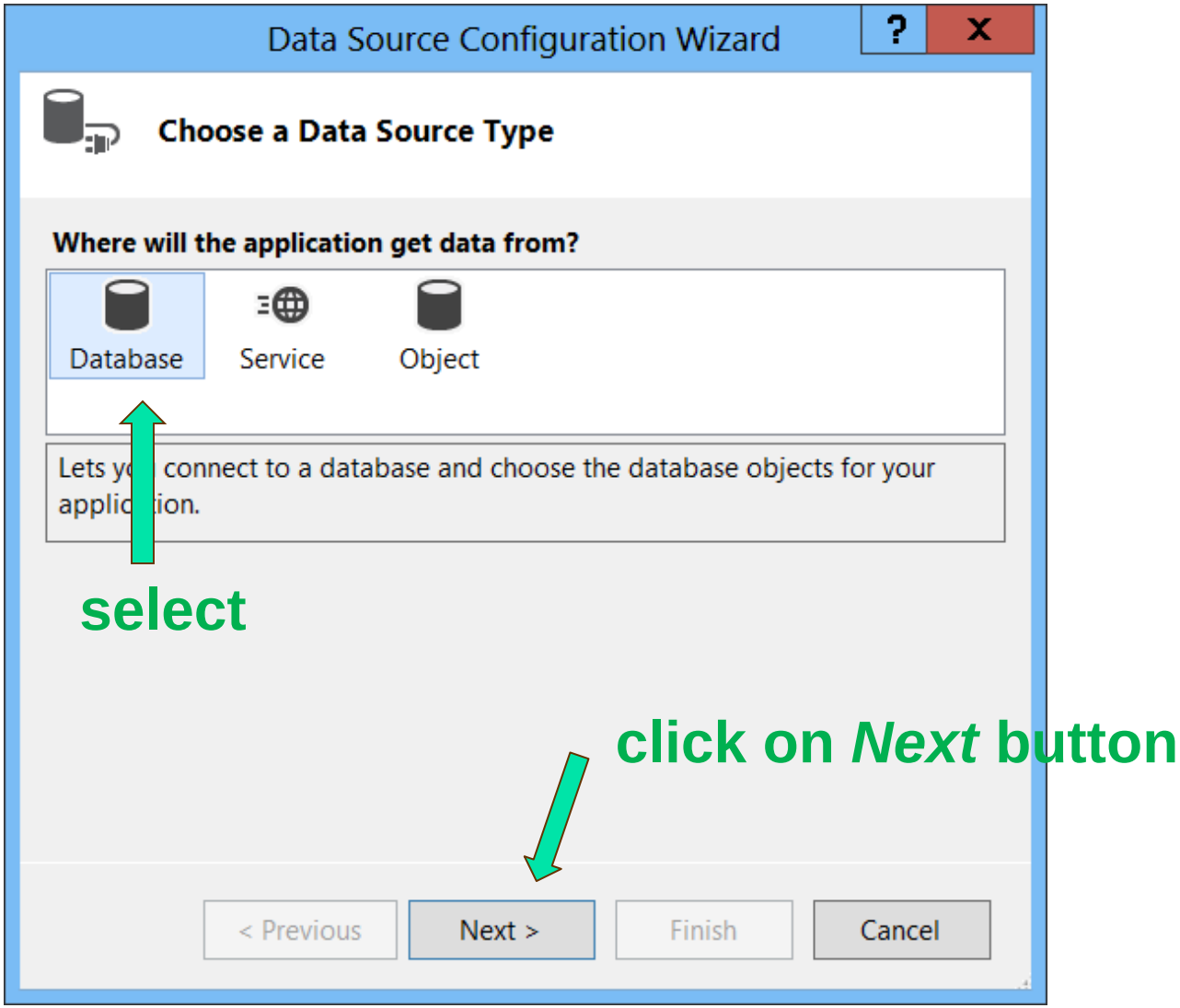
Binding to the Cities Table

Add a BindingSource control to the form. (The control is in the *Data* and *All Windows* Forms group of the Toolbox. It appears in the form's component tray with the name BindingSource1.)

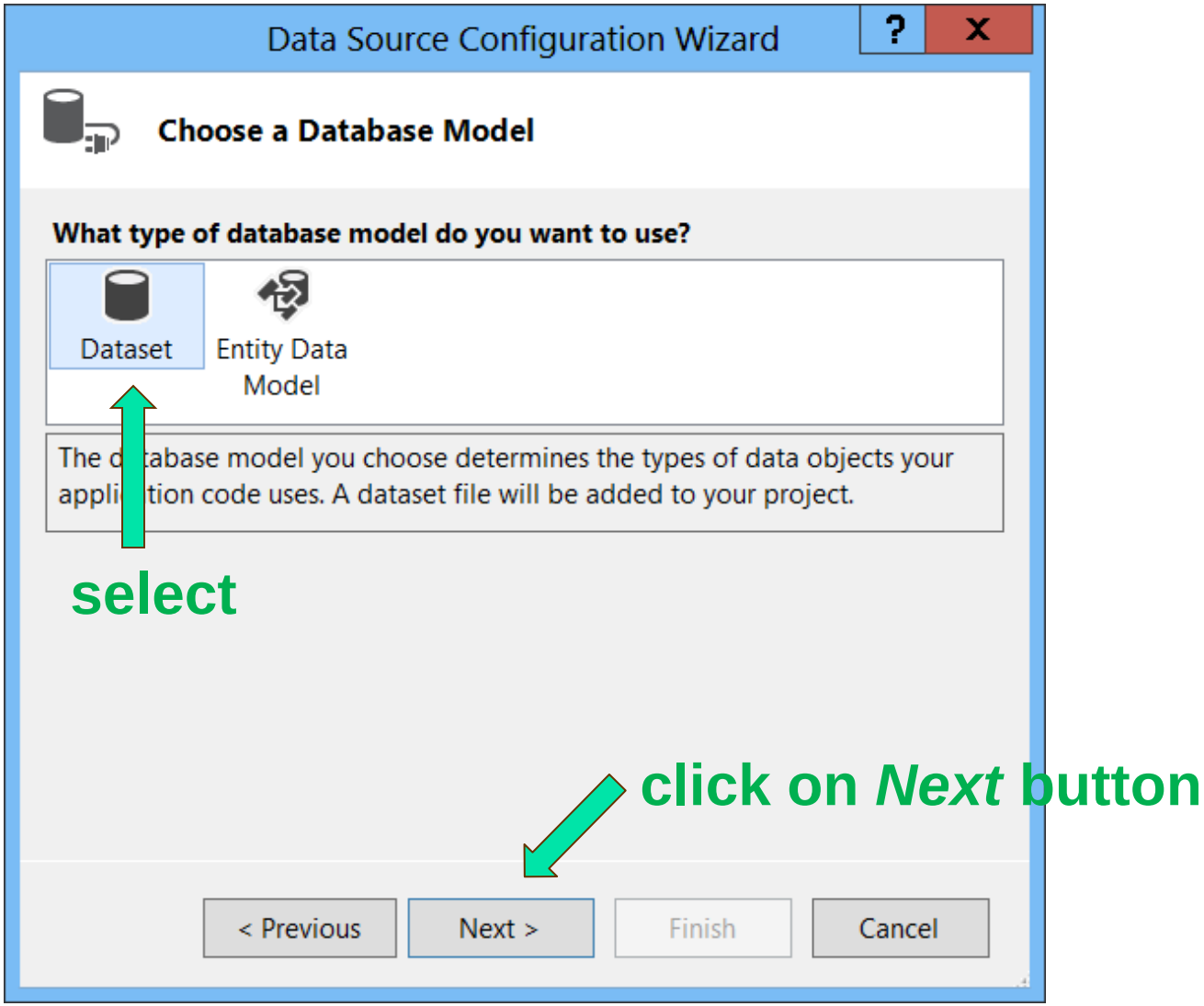
DataSource Property of BindingSource1



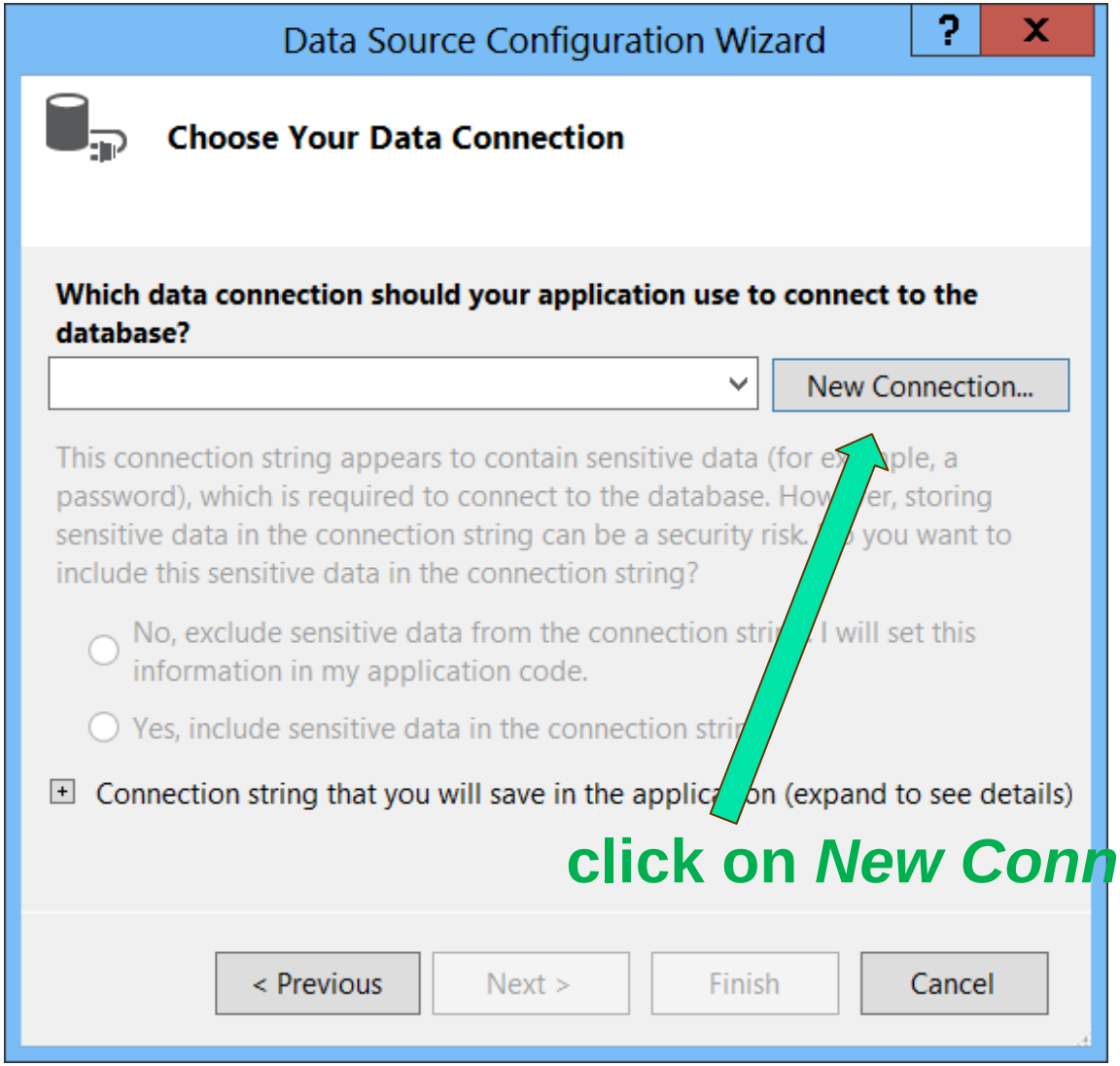
Choose Data Source Type



Choose Database Model

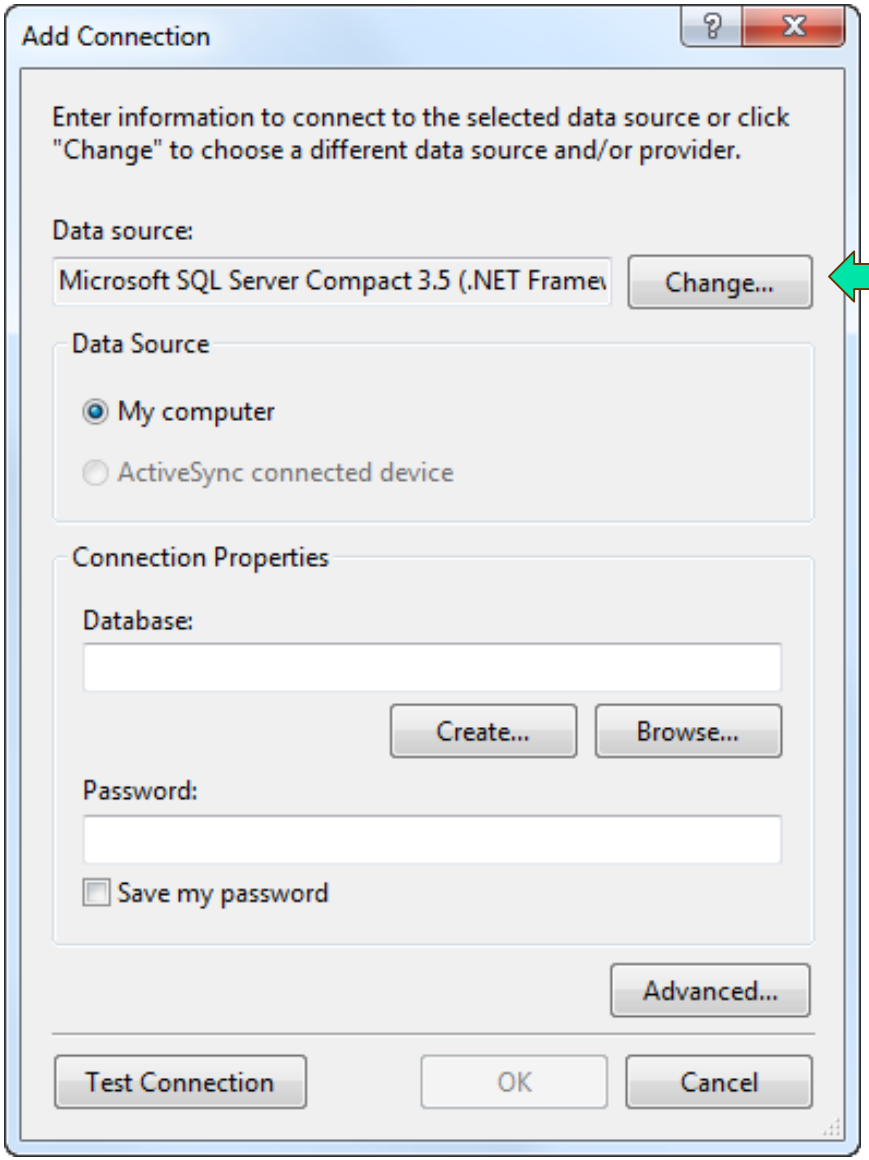


Choose Data Connection



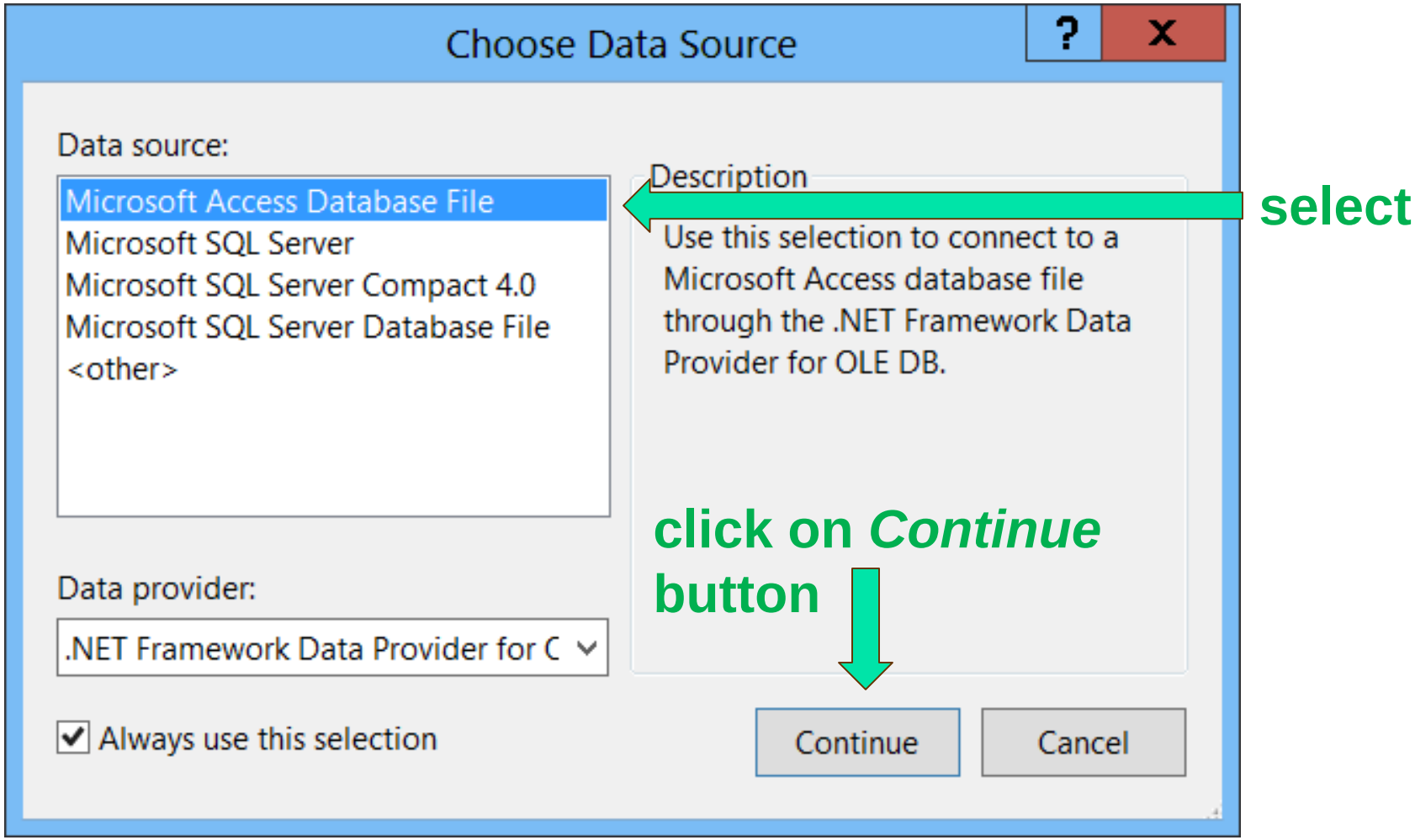
click on *New Connection* button

Add Connection Dialog Box

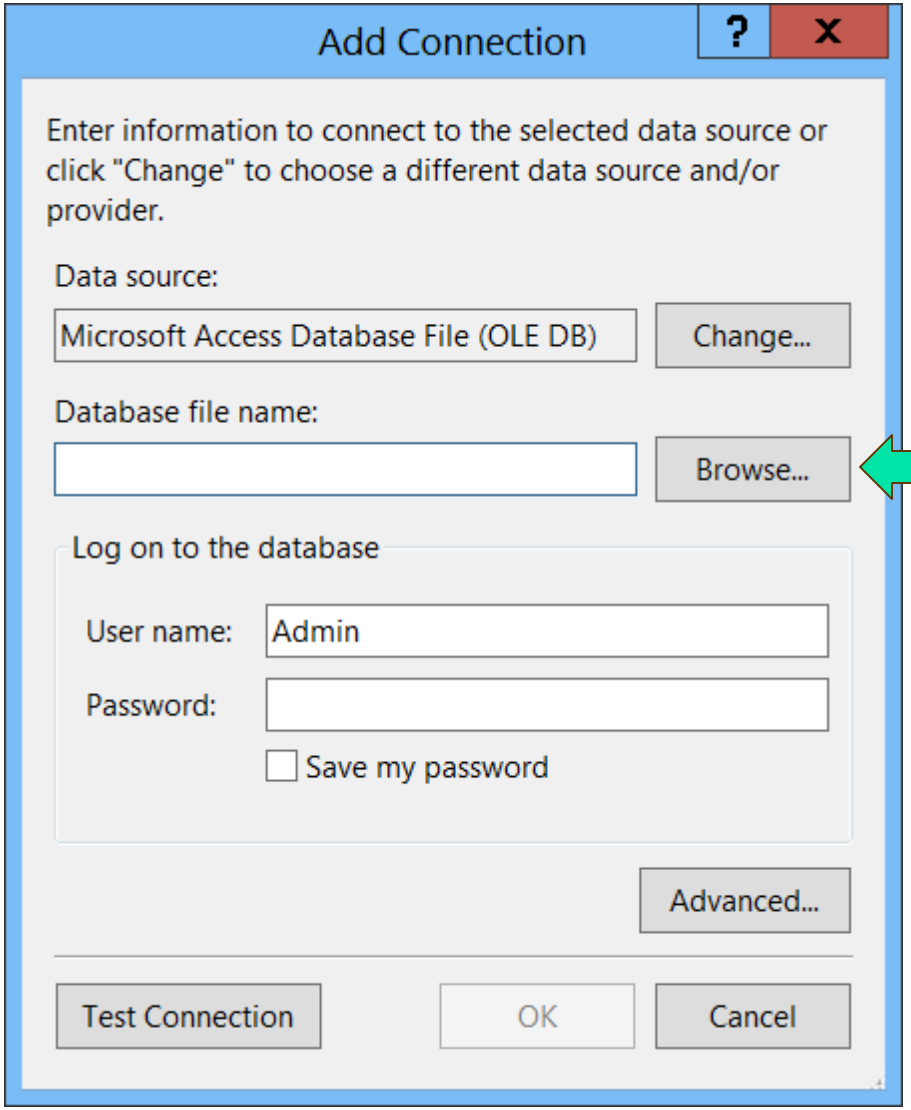


click on *Change* button

Change Data Source Box

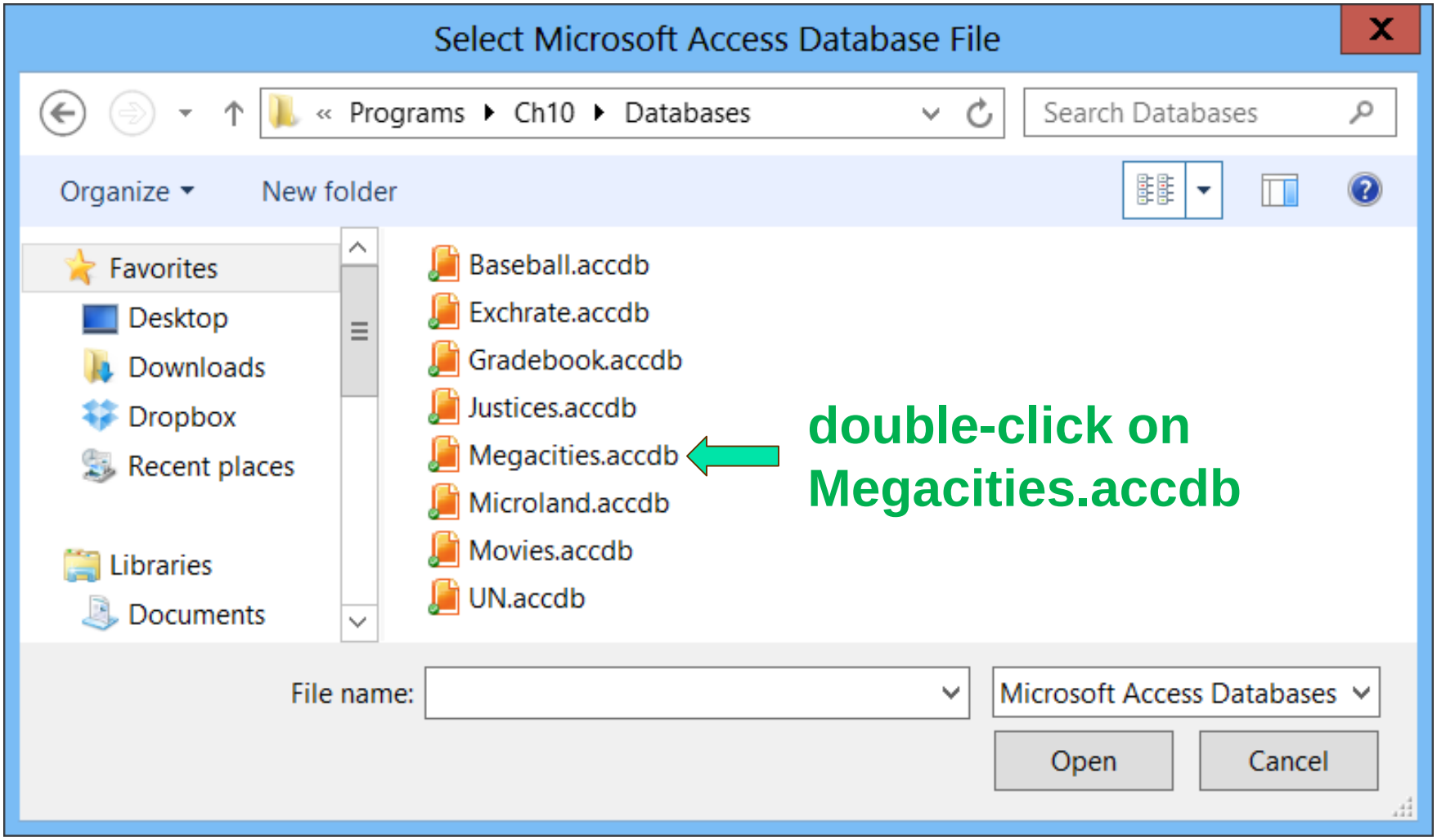


Add Connection Dialog Box



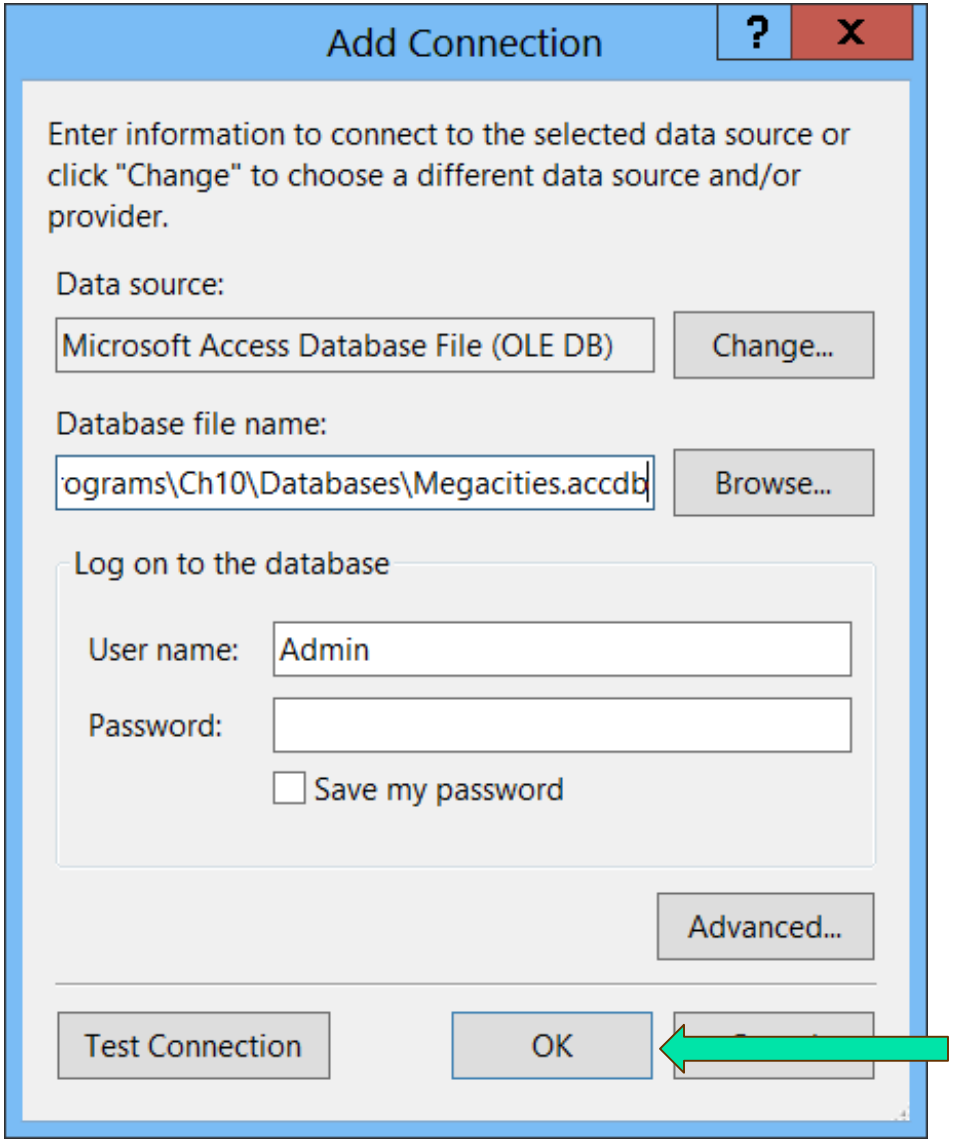
click on *Browse* button

Select Database File



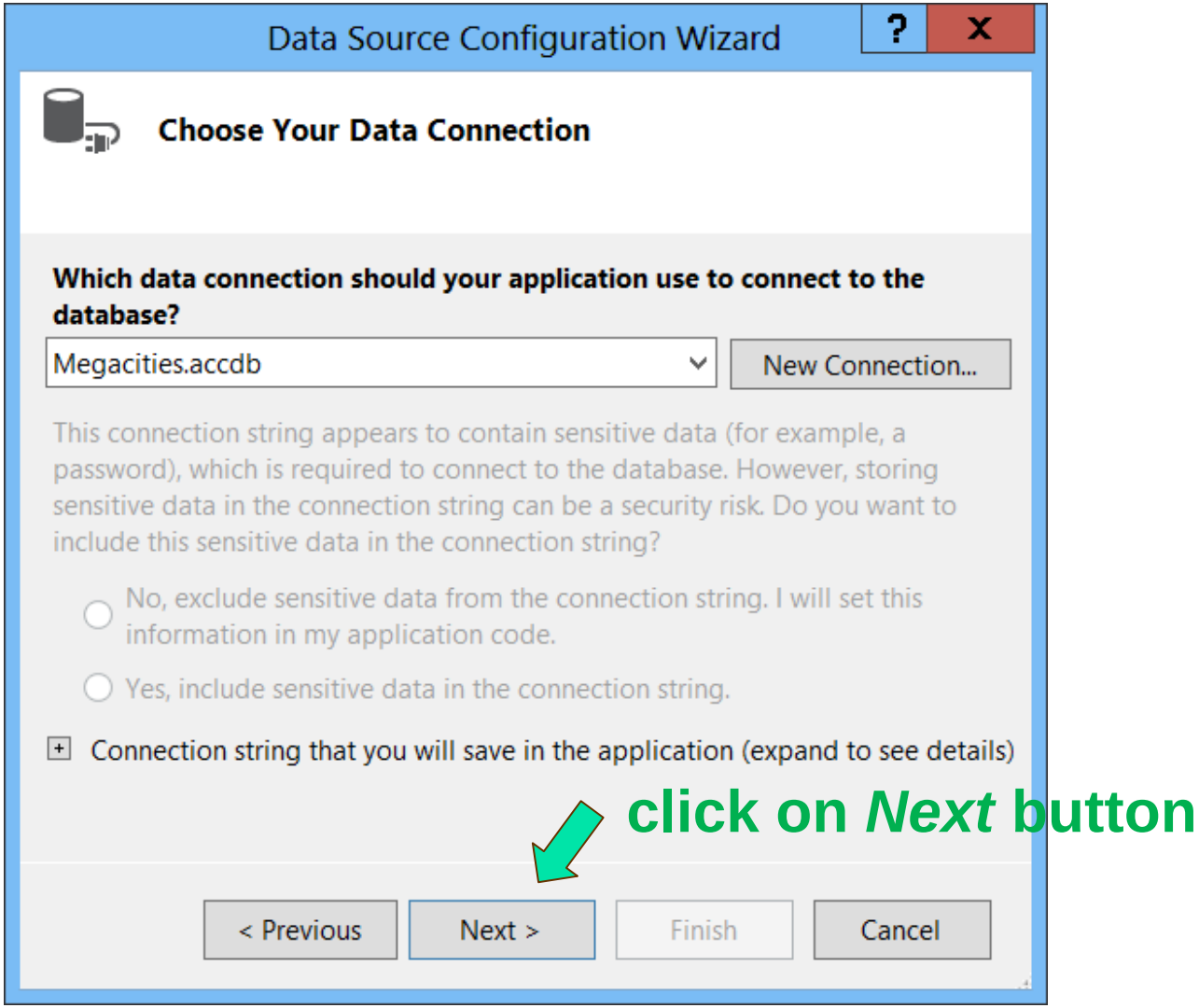
double-click on Megacities.accdb

Add Connection Dialog Box

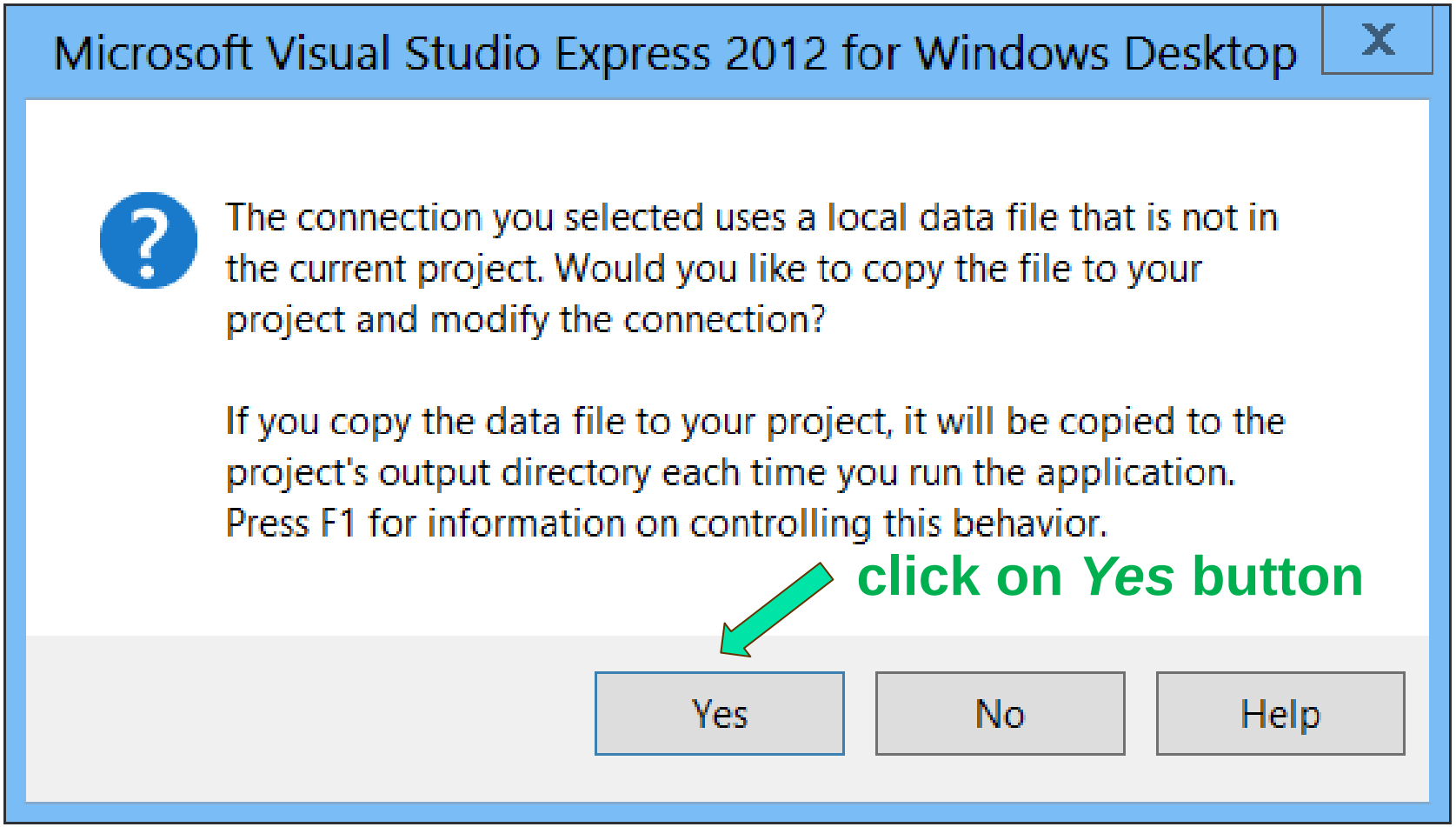


click on *OK* button

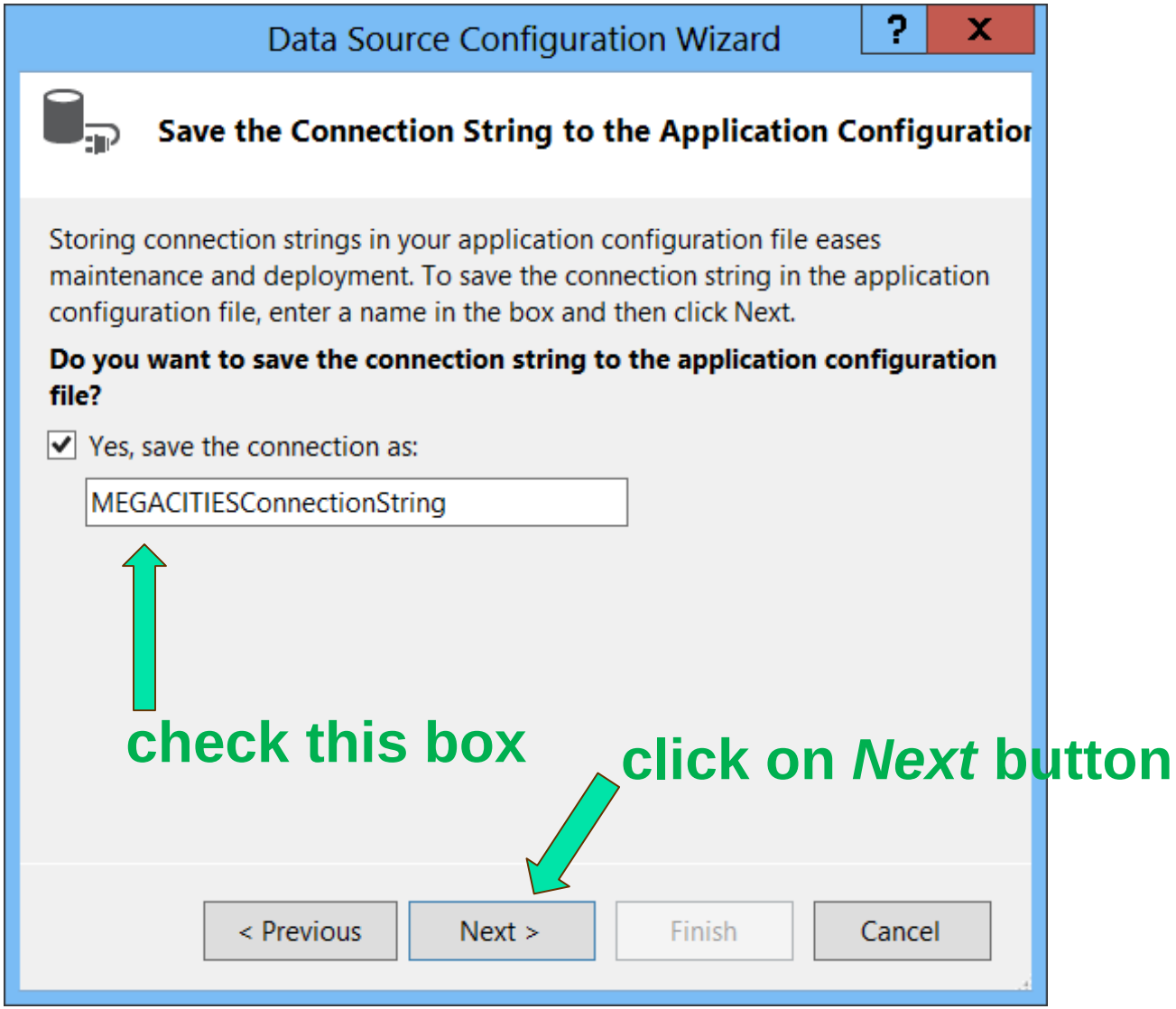
Choose Data Connection



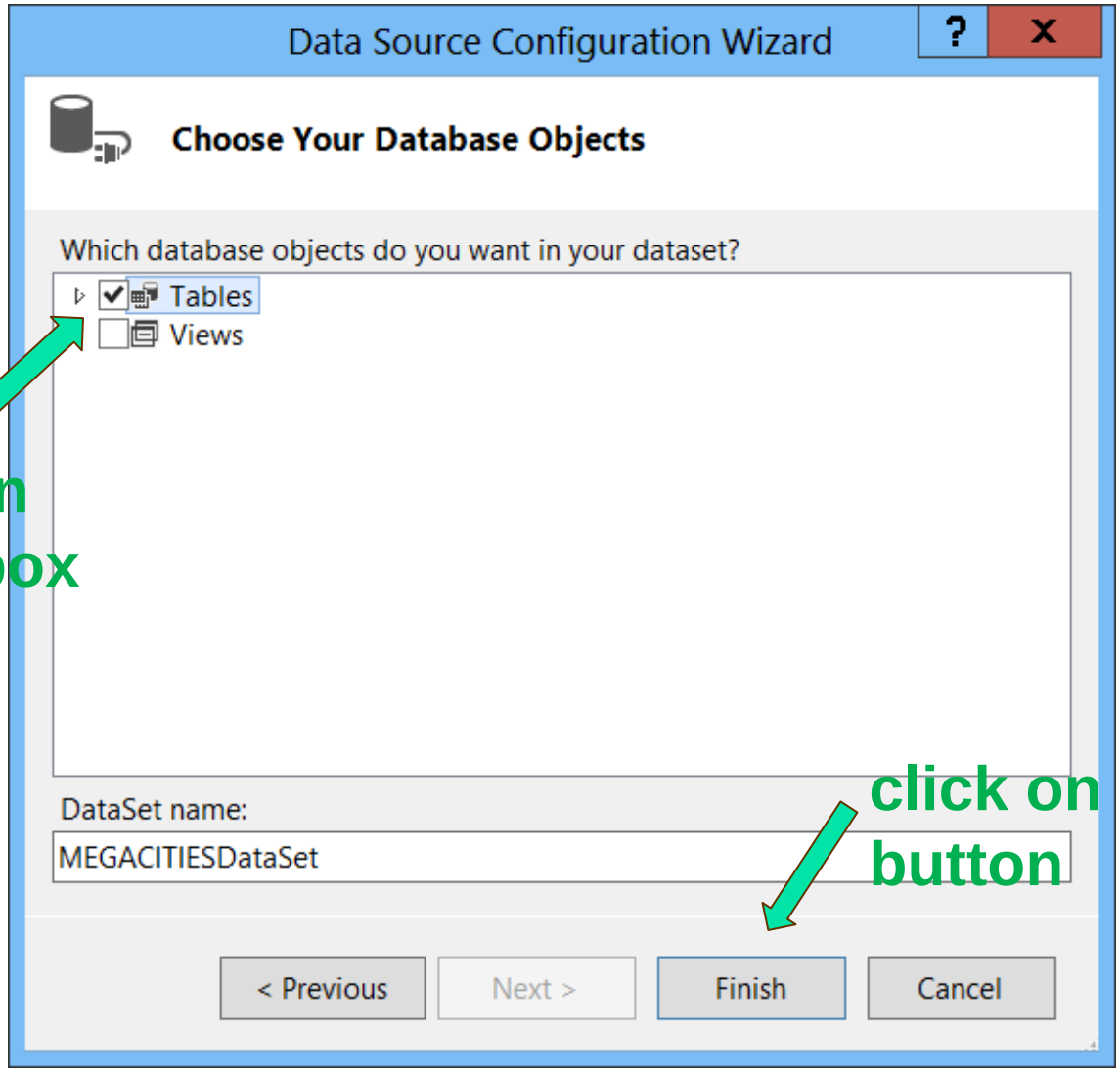
Copy File to Program



Save to File


Data Source Configuration Wizard
Save the Connection String to the Application Configuratior
Storing connection strings in your application configuration file eases maintenance and deployment. To save the connection string in the application configuration file, enter a name in the box and then click Next.
Do you want to save the connection string to the application configuration file?
Yes, save the connection as:
MEGACITIESConnectionString
check this box
click on *Next* button
< Previous
Next >
Finish
Cancel

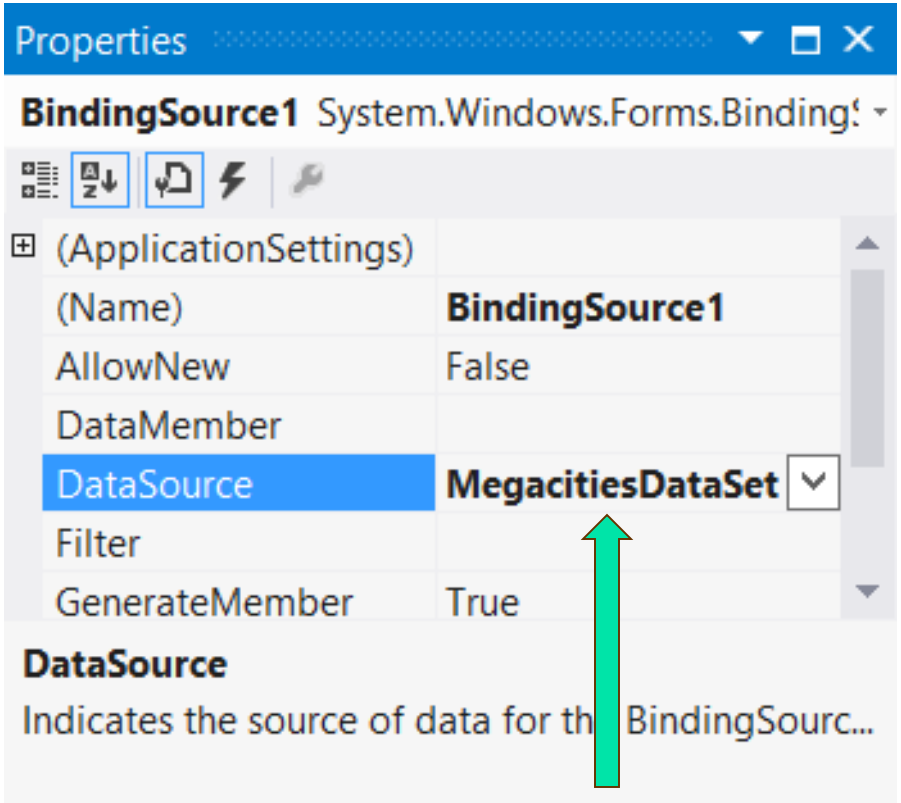
Choose Database Objects

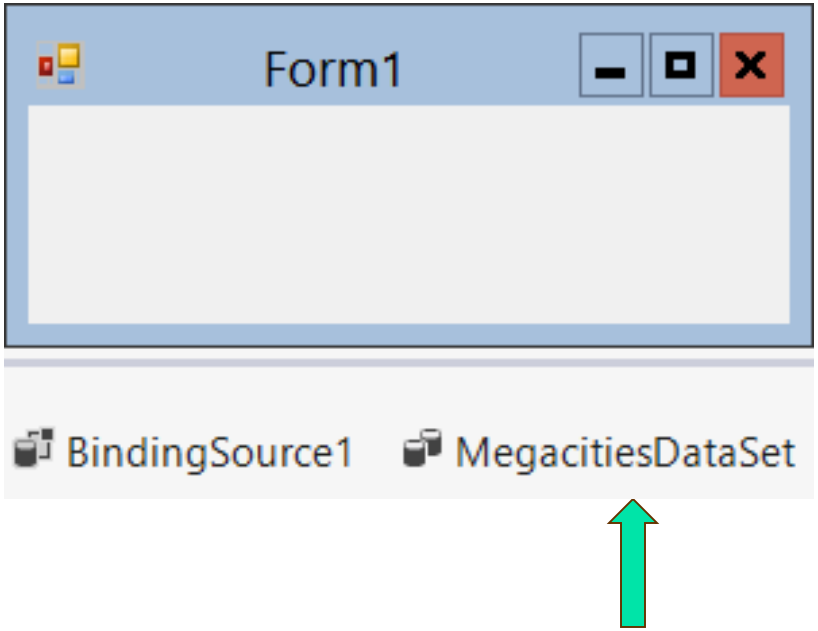


check on Tables box

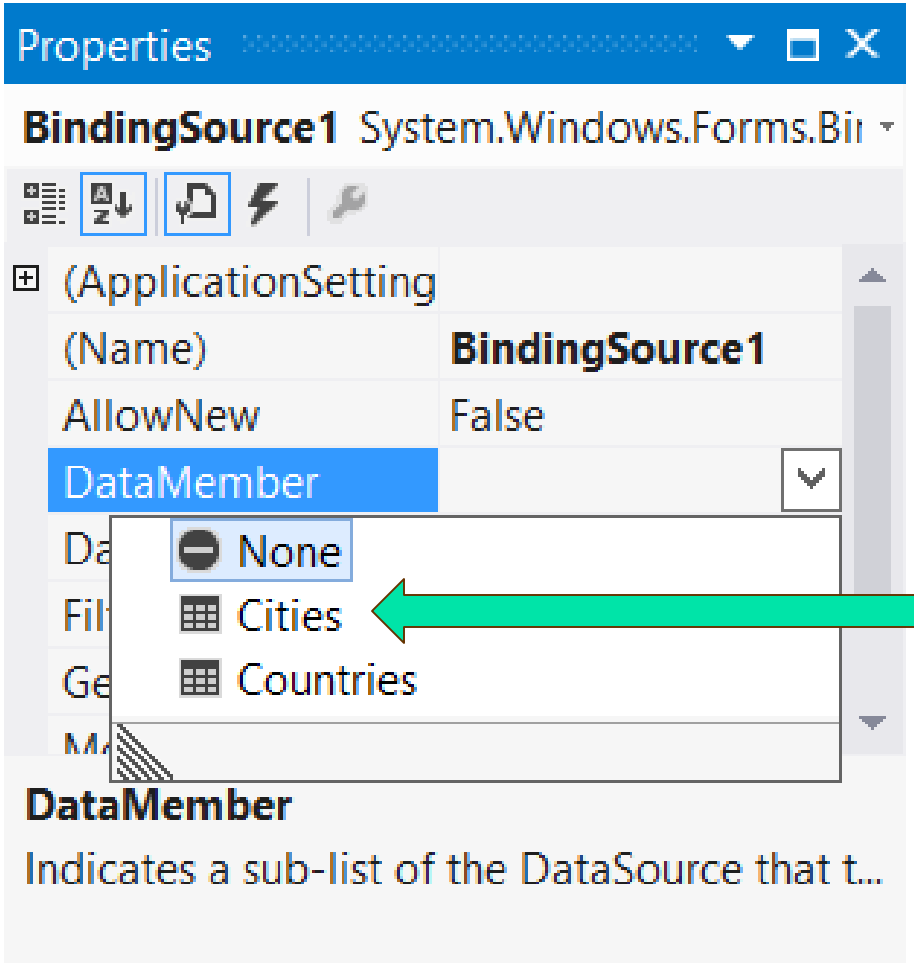
click on *Finish* button

Changes in Properties Window and Form



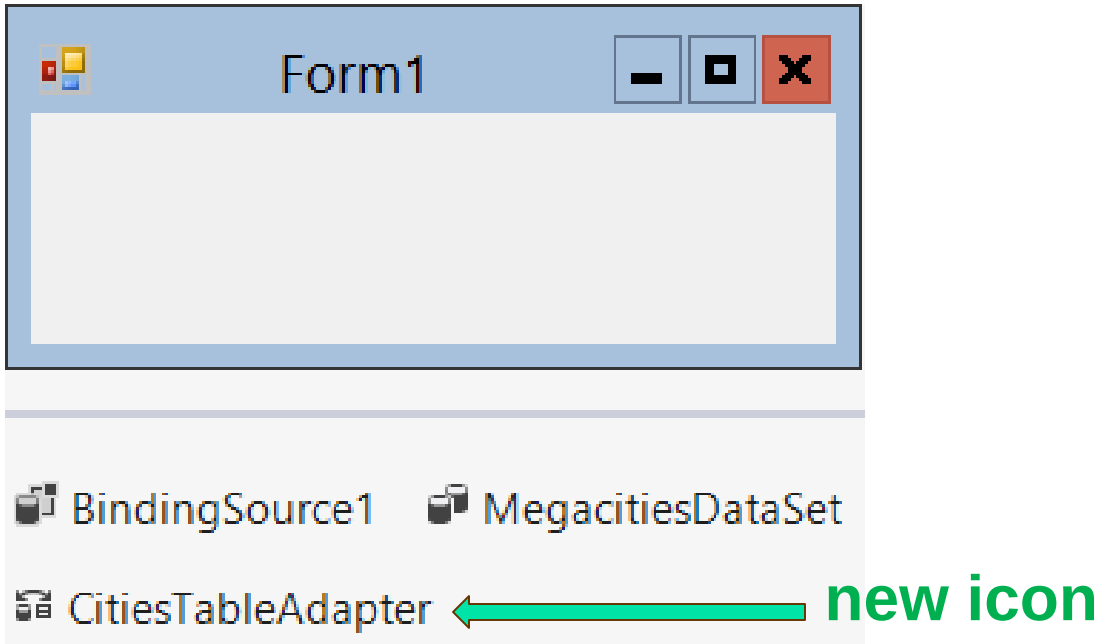


After Clicking on DataMember Down-Arrow



click on *Cities*

C# Generated Items



new code

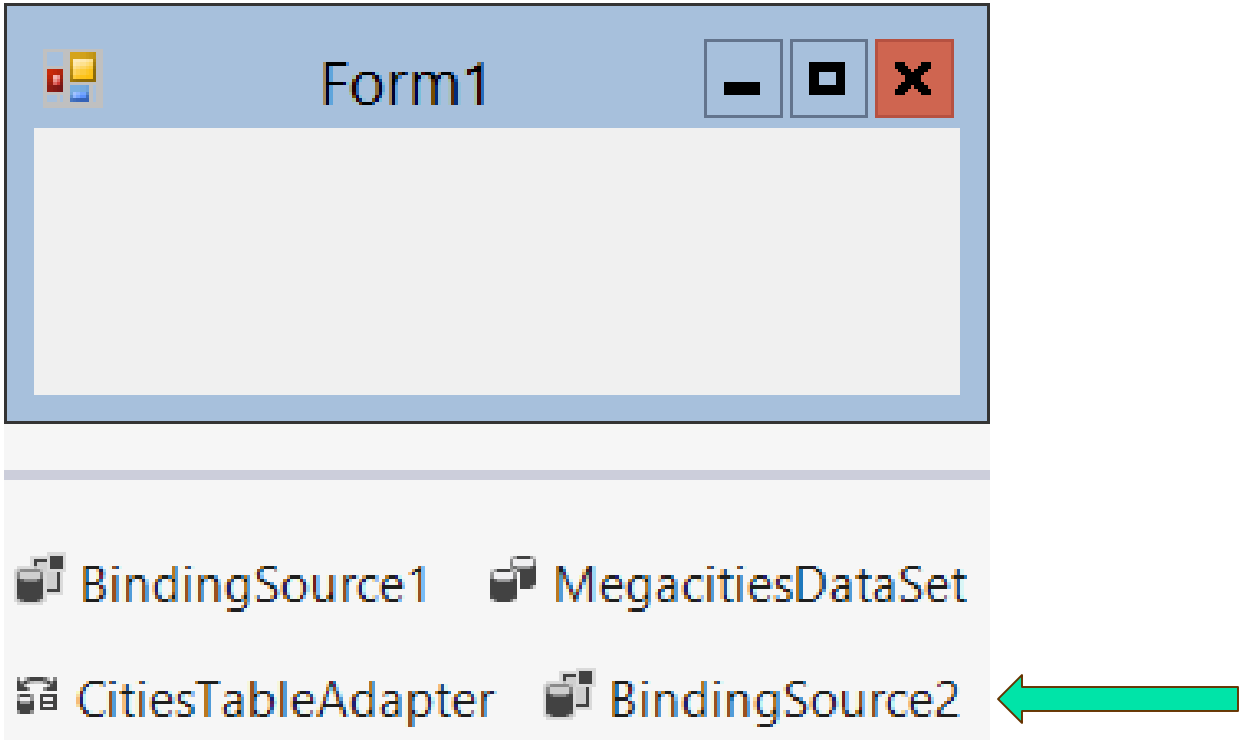
```
private void Form1_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'megacitiesDataSet.Cities‘
    // table. You can move, or remove it, as needed.
    this.citiesTableAdapter.Fill(this.megacitiesDataSet.Cities);
}
```

Binding Complete

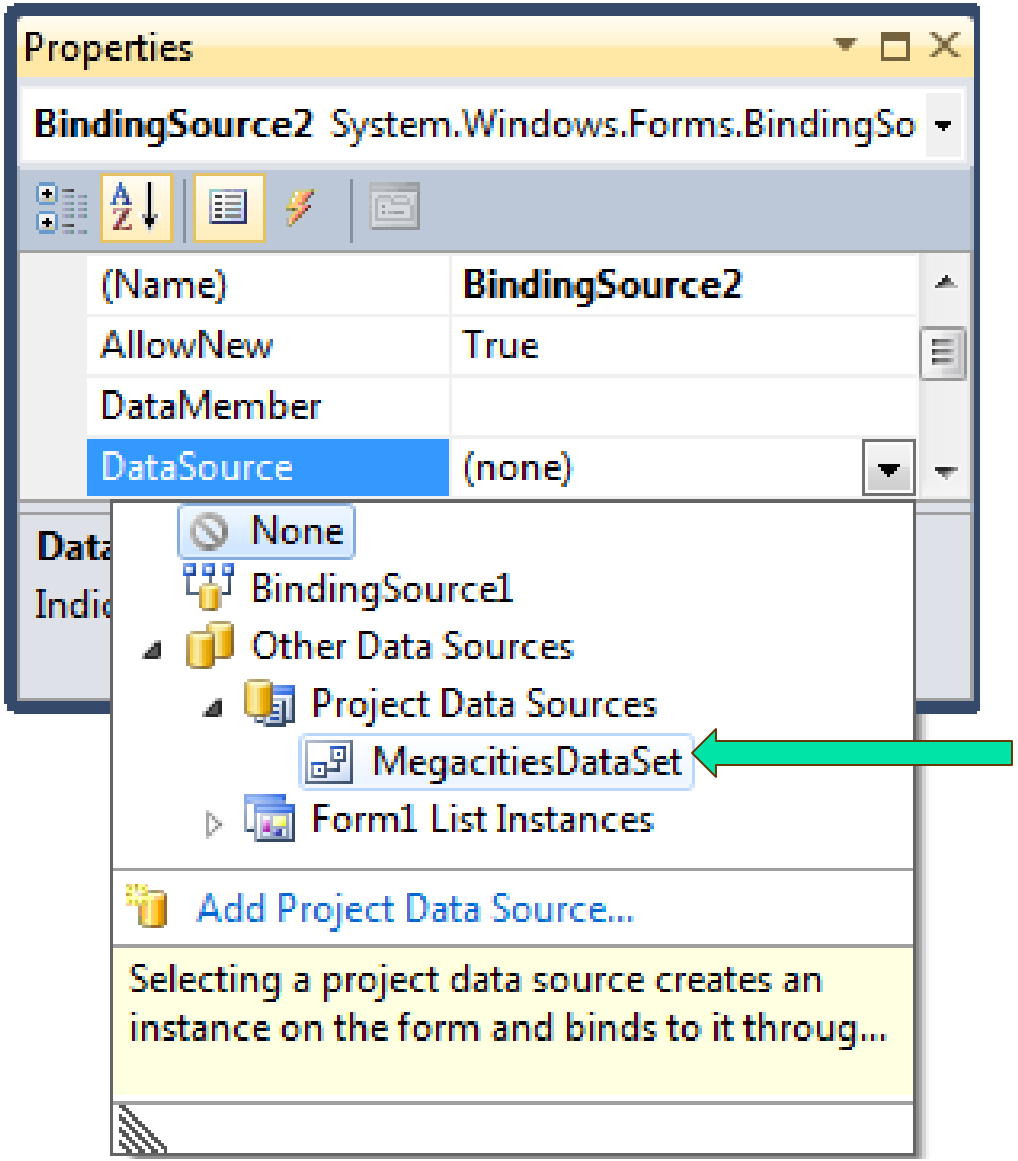
- We are now bound to the Cities table via the MegacitiesDataSet and the CitiesTableAdapter.
- The next four slides show how to bind an additional table.

Connect an Additional Table

Add another BindingSource control to the form.

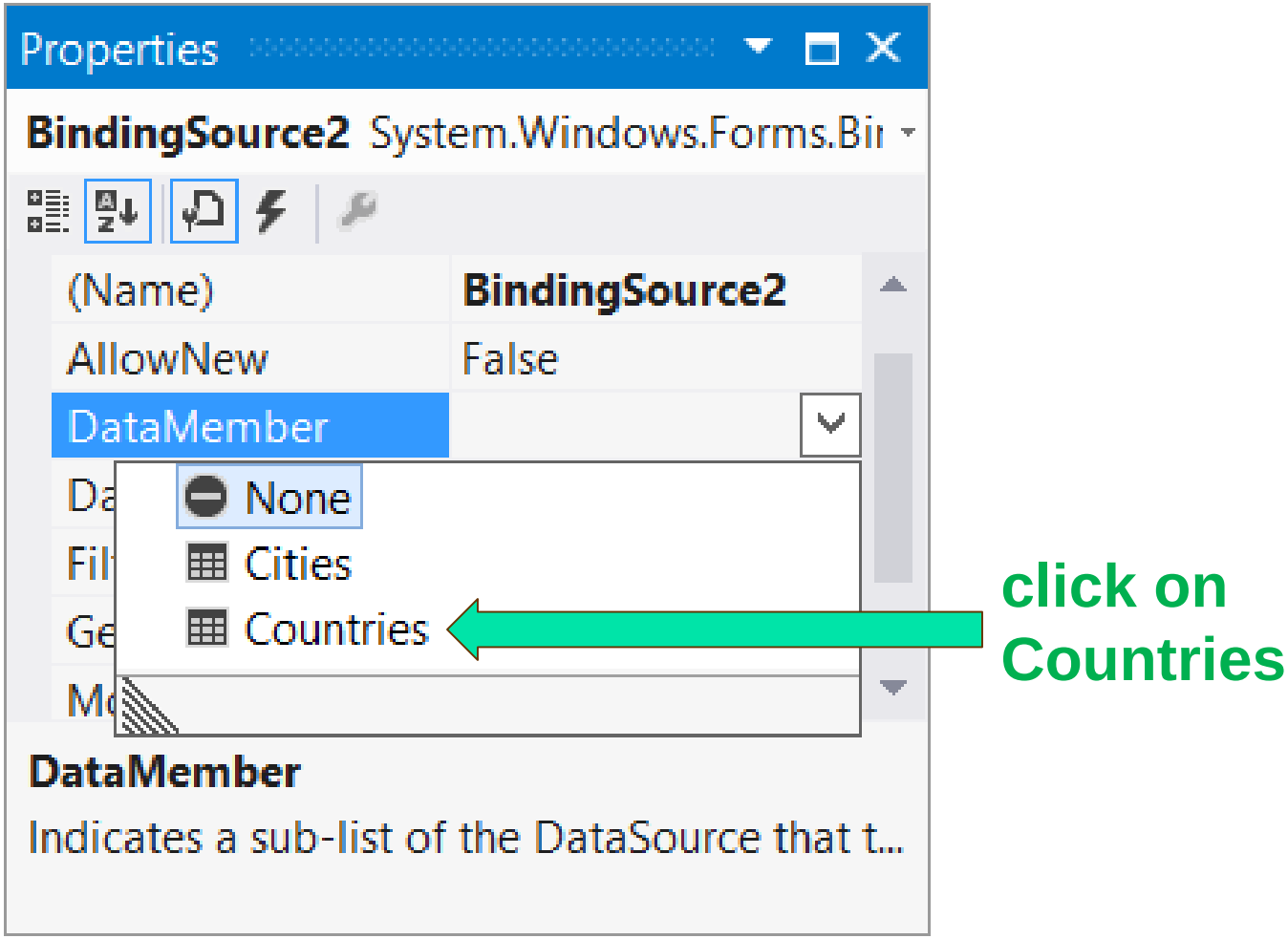


Set DataSource Property

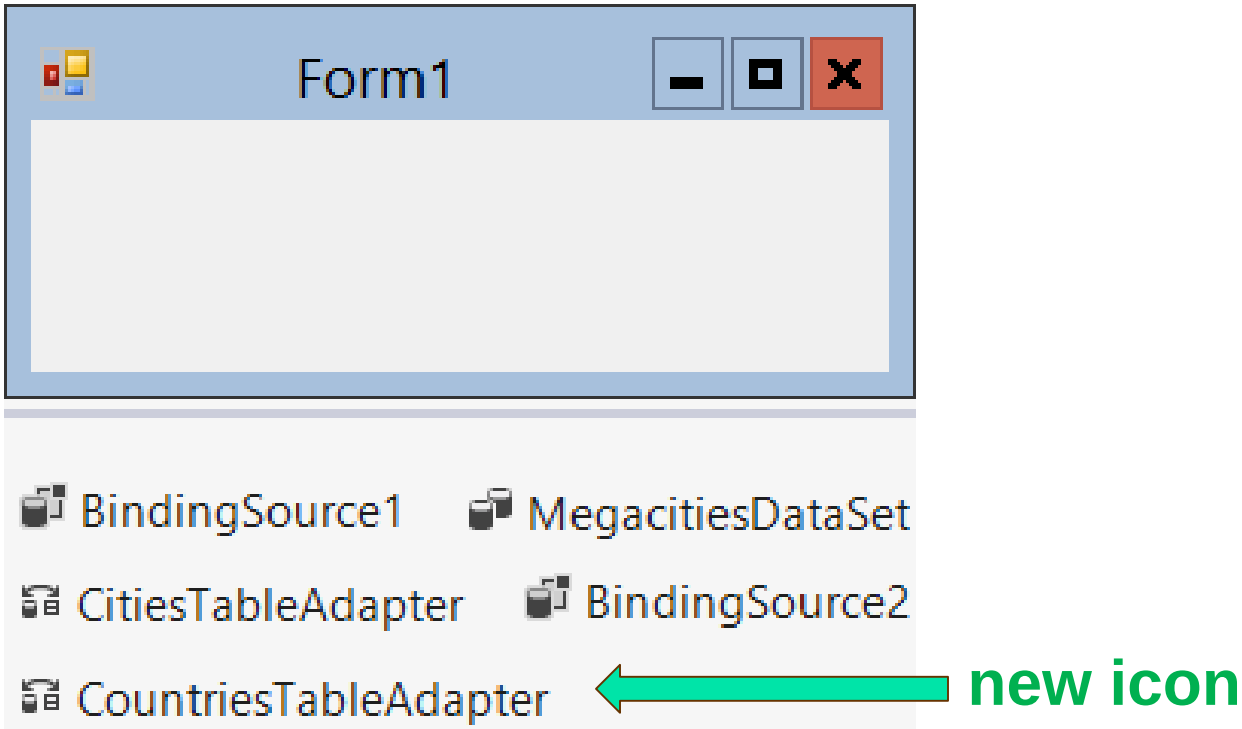


click on MegacitiesDataSet

Set DataMember Property

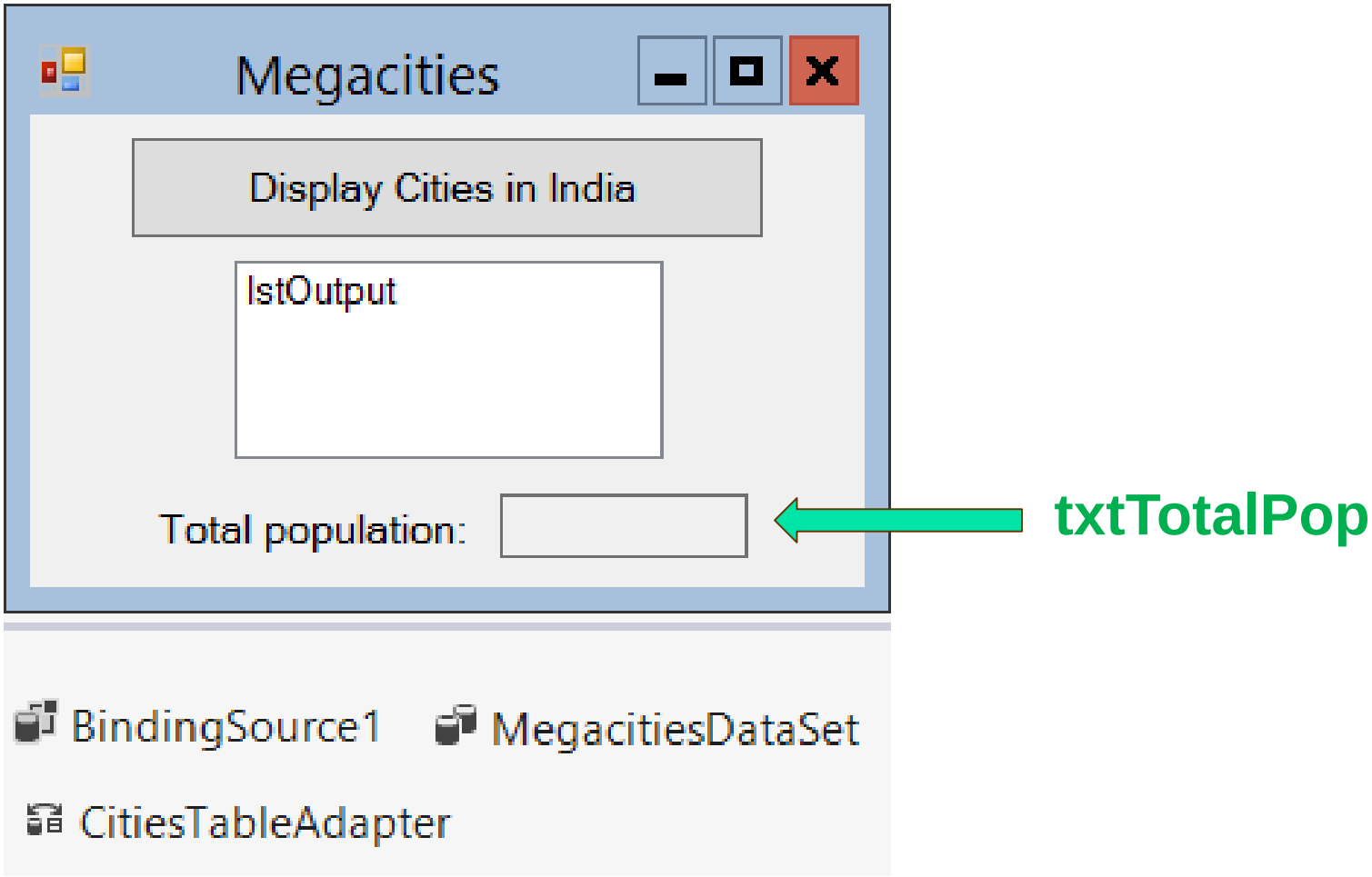


VB Generated Items




```
// Additional code shows in Load event
this.countriesTableAdapter.Fill(this.megacitiesDataSet.Countries);
```

Example 1: Form



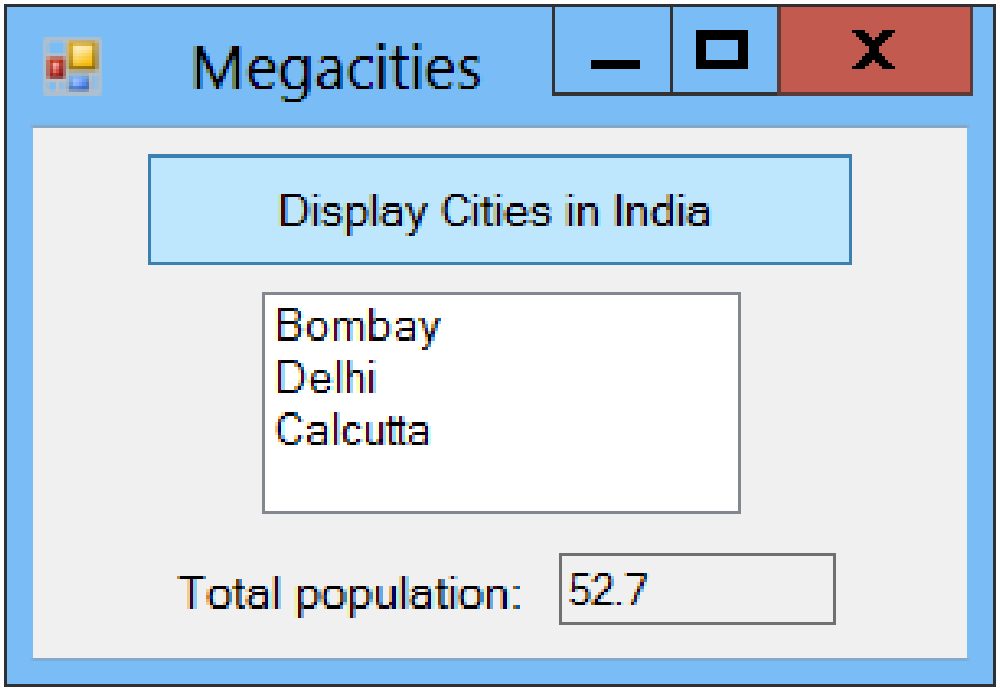
Example 1: Code

```
var myQuery = from city in megacitiesDataSet.Cities
              where city.country == "India"
              select city.name;

listBox1.DataSource = myQuery.ToList();
```

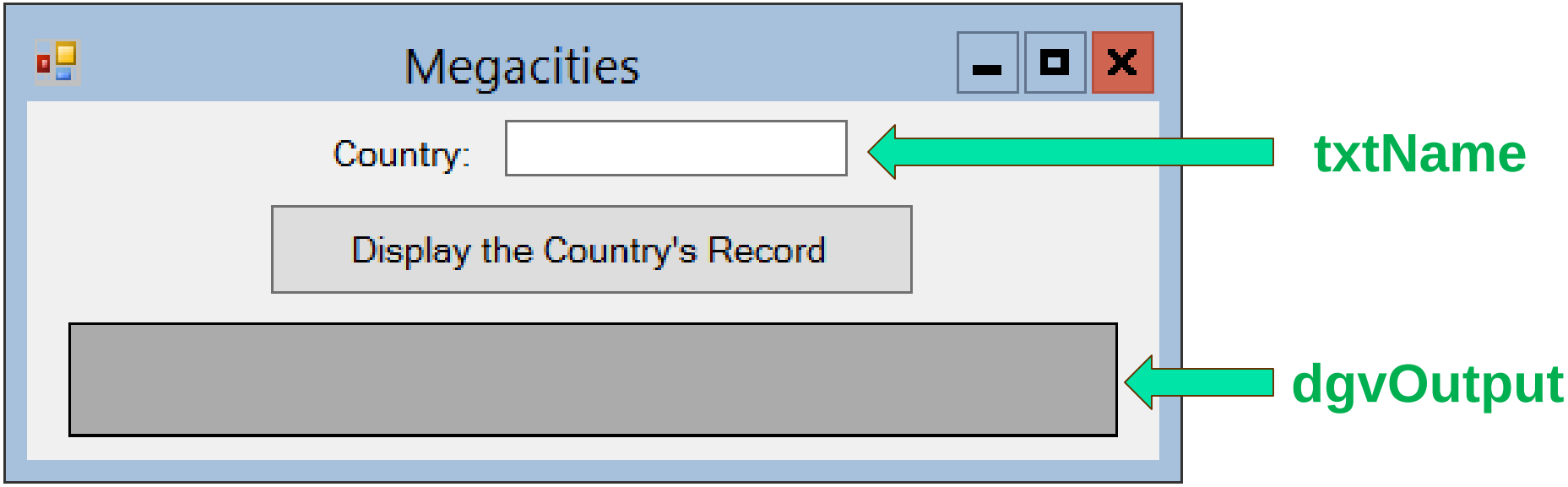
Example 1: Code (continued)

```
var tPopulation = from city in megacitiesDataSet.Cities
                  where city.country == "India"
                  select city.pop2015;

textBox1.Text = tPopulation.Sum().ToString("n2");
```

Example 1: Output



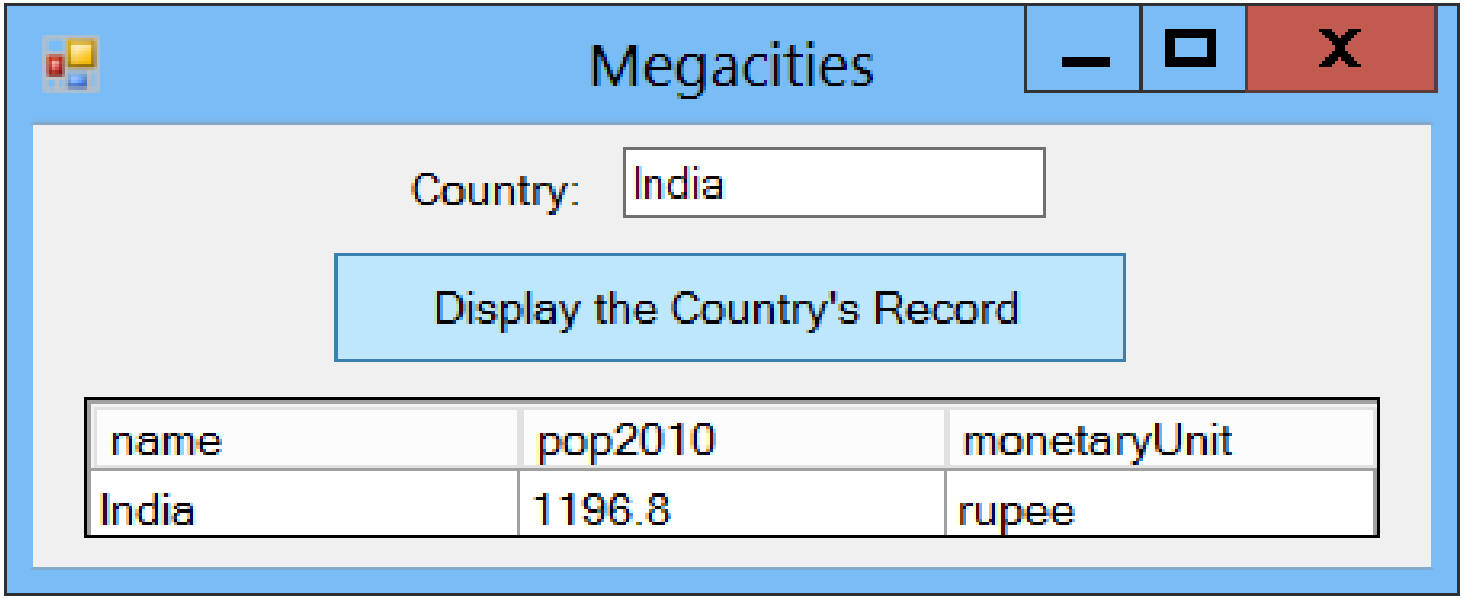
Example 2: Form



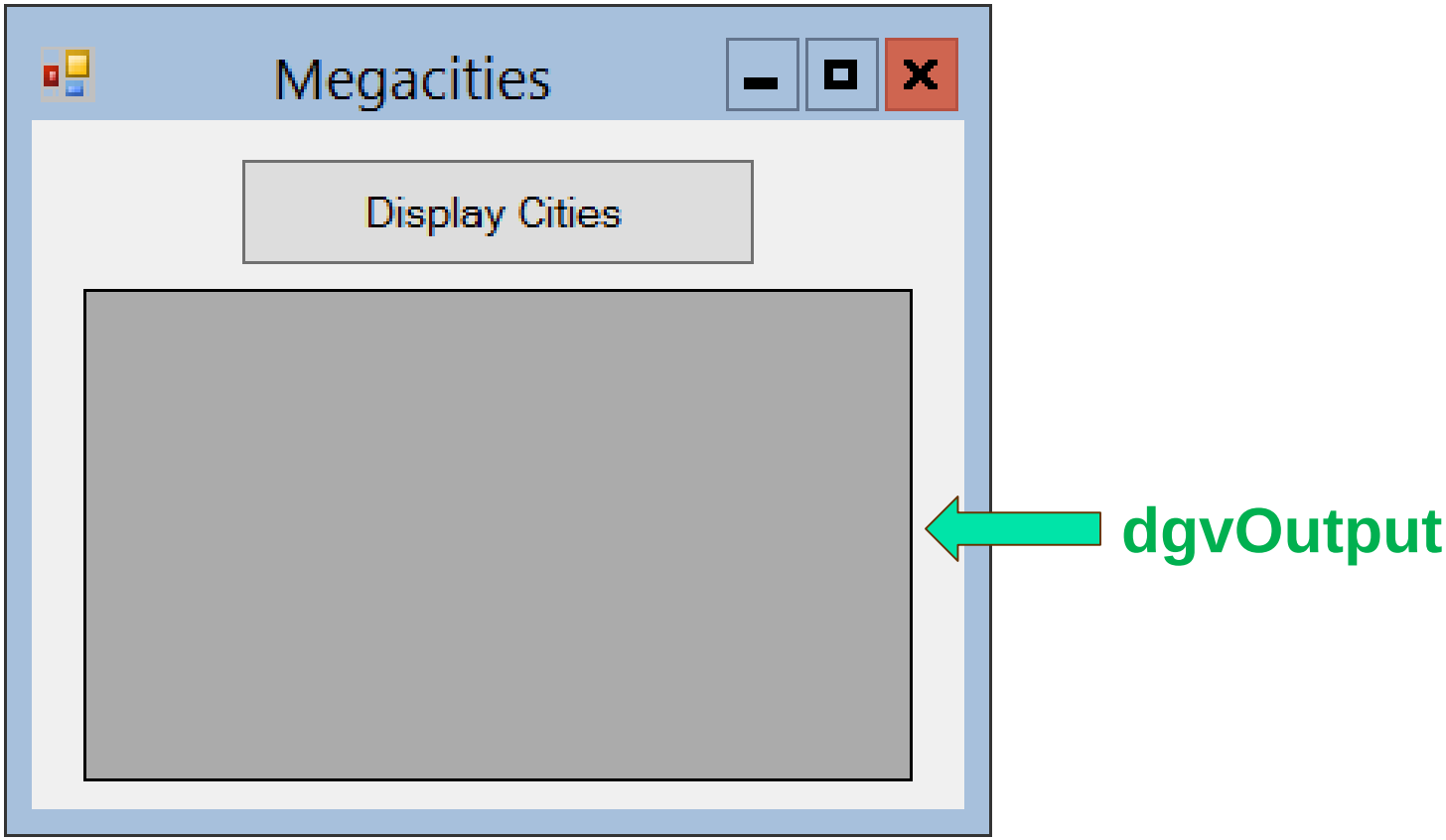
Example 2: Code

```
var cntInfo = from item in megacitiesDataSet.Countries
              where item.name == textBox1.Text
              let cName = item.name
              let cPop = item.pop2010
              let cCurr = item.monetaryUnit
              select new { cName, cPop, cCurr };
if (cntInfo.Count() == 1)
    dataGridView1.DataSource = cntInfo.ToList();
else
    MessageBox.Show("Country not found!");
```

Example 2: Output



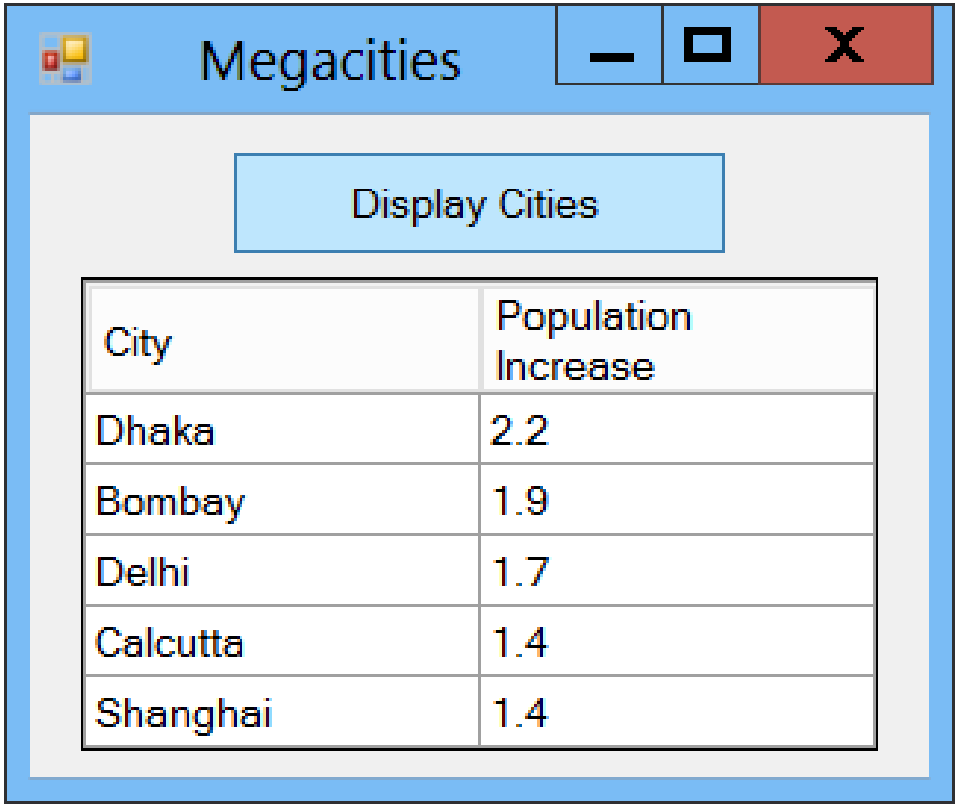
Example 3: Form



Example 3: Code

```
var ctyInfo = from city in megacitiesDataSet.Cities
              let popInc = city.pop2015 - city.pop2010
              let formattedInc = popInc.ToString("n1")
              where popInc > 1
              orderby popInc descending
              select new { city.name, formattedInc };

dataGridView1.DataSource = ctyInfo.ToList();
dataGridView1.Columns[0].HeaderText = "City";
dataGridView1.Columns[1].HeaderText =
                          "Population Increase";
```

Example 3: Output



City	Population Increase
Dhaka	2.2
Bombay	1.9
Delhi	1.7
Calcutta	1.4
Shanghai	1.4

Primary Keys

- A **primary** key is used to uniquely identify each record.
- Databases of student enrollments in a college usually use a field of student ID numbers as the primary key.
- Why wouldn't names be a good choice as a primary key?

Primary Key Fields

- Specified when database is created.
- Every record must have an entry in the primary-key field.
- Two records cannot have the same entry in the primary-key field.
- This pair of requirements is called the **Rule of Entity Integrity**.

Two or More Tables

- When a database contains two or more tables, the tables are usually related.
- For instance, the two tables Cities and Countries are related by their country and name fields.
- Notice that every entry in Cities.country appears uniquely in Countries.name and Countries.name is a primary key.
- We say that Cities.country is a **foreign key** of Countries.name.

Sample Table

Cities Table

name	country	pop2010	pop2015
Bombay	India	20.1	22
Buenos Aires	Argentina	13.1	13.4
Calcutta	India	15.6	17
Delhi	India	17	18.7
Dhaka	Bangladesh	14.8	17
Mexico City	Mexico	19.5	20.2
New York	USA	19.4	20
Sao Paulo	Brazil	19.6	20.1
Shanghai	China	15.8	17.2
Tokyo	Japan	36.1	36.4

Countries Table

name	pop2010	monetaryUnit
Argentina	41.9	peso
Bangladesh	152.6	raka
Brazil	195.2	real
China	1379.7	yuan
India	1196.8	rupee
Indonesia	258.5	rupiah
Japan	129	yen
Mexico	117.4	peso
Pakistan	184.2	rupee
USA	299	dollar

Foreign Keys

- Foreign keys can be specified when a table is first created. Visual C# will insist on the **Rule of Referential Integrity.**
- This Rule says that each value in the foreign key must also appear in the primary key of the other table.

Join

- A foreign key allows Visual C# to link (or **join**) two tables from a relational database
- When the two tables Cities and Countries from Megacities.accdb are joined based on the foreign key Cities.country, the result is the table in the next slide.
- The record for each city is expanded to show its country's population and its monetary unit.

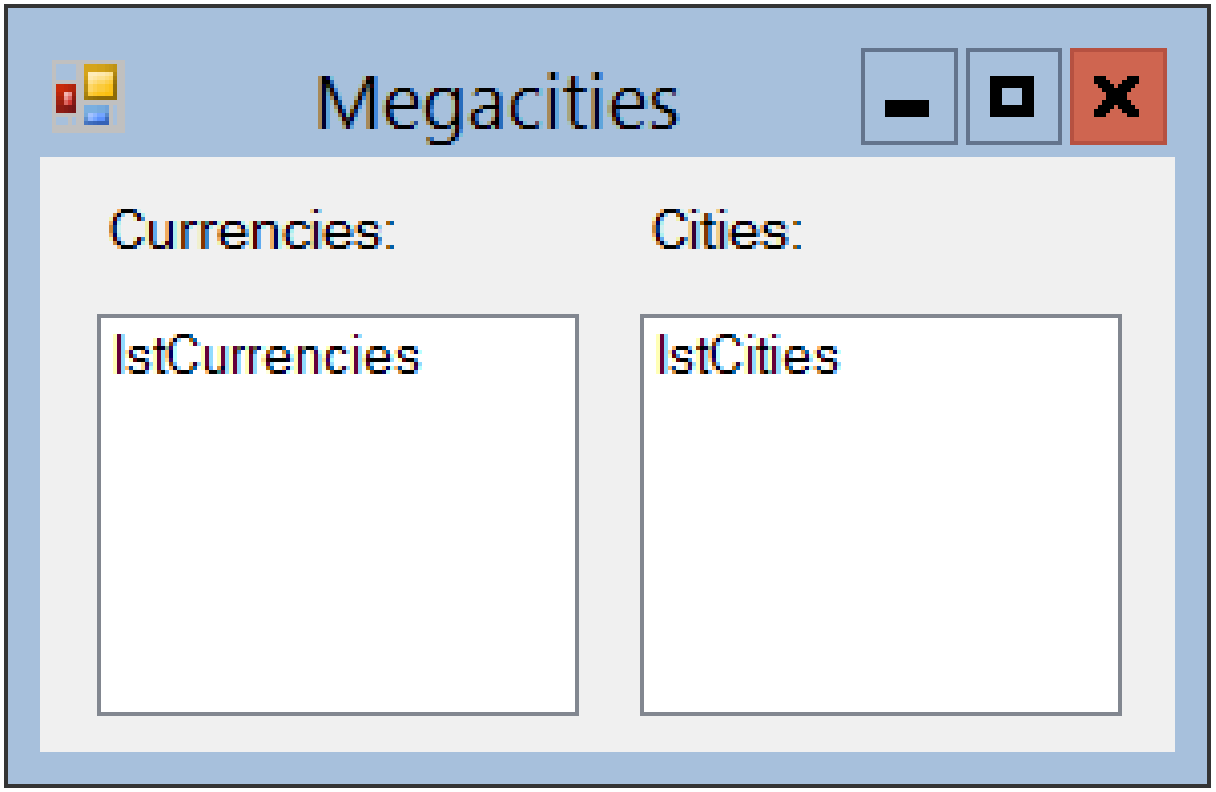
A Join of Two Tables

Cities.name	Cities. country	Cities. pop2010	Cities. pop2015	Countries. name	Countries. pop2010	Countries. monetaryUnit
Bombay	India	20.1	22.0	India	1196.8	rupee
Buenos Aires	Argentina	13.1	13.4	Argentina	41.9	peso
Calcutta	India	15.6	17.0	India	1196.8	rupee
Delhi	India	17.0	18.7	India	1196.8	rupee
Dhaka	Bangladesh	14.8	17.0	Bangladesh	152.6	rupee
Mexico City	Mexico	19.5	20.2	Mexico	117.4	peso
New York	USA	19.4	20.0	USA	310.1	dollar
Sao Paulo	Brazil	19.6	20.1	Brazil	195.2	real
Shanghai	China	15.8	17.2	China	1379.7	yuan
Tokyo	Japan	36.1	36.4	Japan	129.0	yen

Beginning of Query to Join the Two Tables from Megacities

```
var myquery = from city in megacitiesDataSet.Cities
              join country in megacitiesDataSet.Countries
              on city.country equals country.name
```

Example 6: Form



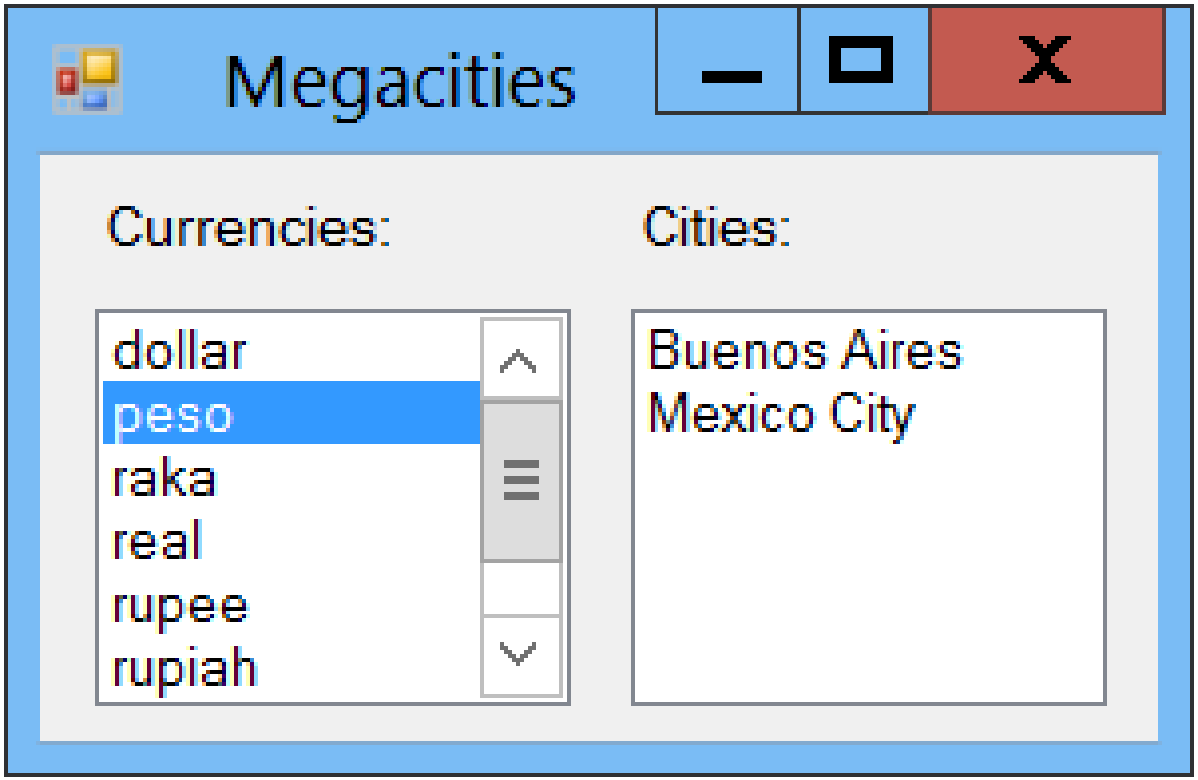
Example 6: Code for Load Event

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

    var query = from country in megacitiesDataSet.Countries
                orderby country.name ascending
                select country.monetaryUnit;

    listBox1.DataSource = query.Distinct().ToList();
    listBox1.SelectedItem = null;
}
```

Example 6: Code for SelectedIndexChanged Event

```
private void listBox1_SelectedIndexChanged(object
                                           sender, EventArgs e)
{
    var myquery = from city in megacitiesDataSet.Cities
                  join country in
                        megacitiesDataSet.Countries
                  on city.country equals country.name
                  where country.monetaryUnit ==
                                          listBox1.Text
                  orderby city.name ascending
                  select city.name;
    listBox2.DataSource = myquery.ToList();
}
```

Example 6: Sample Output



Principles of Database Design

- Data should usually be stored in their smallest parts.
- Avoid redundancy.
- Avoid tables with intentionally blank entries.
- Strive for table cohesion.
- Avoid fields whose values can be calculated from existing fields.