THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

DEPARTMENT OF
COMPUTER SCIENCE
Te Tari Rorohiko

2025

Contributors

J. Turner

R. Mercado

V. Moxham-Bettridge

J. Kasmara

© 2025 University of Waikato. All rights reserved. No part of this book may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without prior consent of the Department of Computer Science, University of Waikato.
The course material may be used only for the University's educational purposes. It includes extracts of copyright works copied under copyright licences. You may not copy or distribute any part of this material to any other person, and may print from it only for your own use. You may not make a further copy for any other purpose. Failure to comply with the terms of this warning may expose you to legal action for copyright infringement and/or disciplinary action by the University.

THE SOFTWARE DEVELOPMENT LIFECYCLE: IMPLEMENTATION

Last week you began building the user interface for the Parkway Commute project program in Visual Studio using the C# programming language. Today's exercises will familiarise you with the backend of implementation with a focus on databases.

Backend Development

In our previous session we discussed backend and frontend development. Recall that backend programming is developing the part of the software that the user can't see. In most modern applications today this refers to the functionality of the program. Note that when most people talk about backend development they usually do this in reference to web development. However, this concept can be more generally applied.

For example, consider a simple calculator application. The front end of the application contains buttons for the user to input an equation, the backend will perform the calculation and return the answer for display.

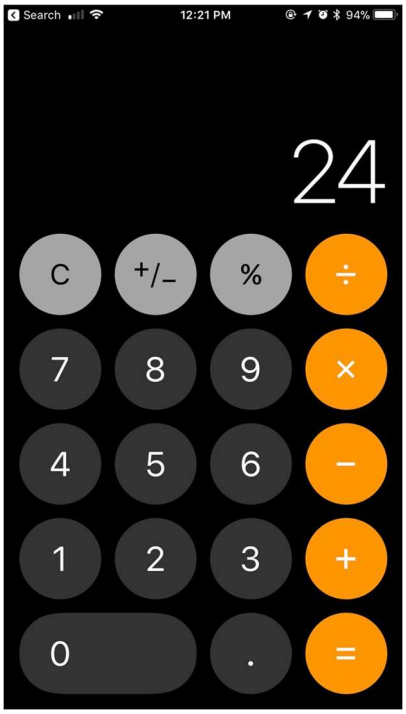


Figure 17: Calculator Application

Most modern applications use what we call the Model-View-Controller software design pattern, or MVC pattern. This separates the software into three clear components: the view which is essentially the user interface of the system; the controller, which sends information between the frontend and backend; and the model, which represents the backend functionality.

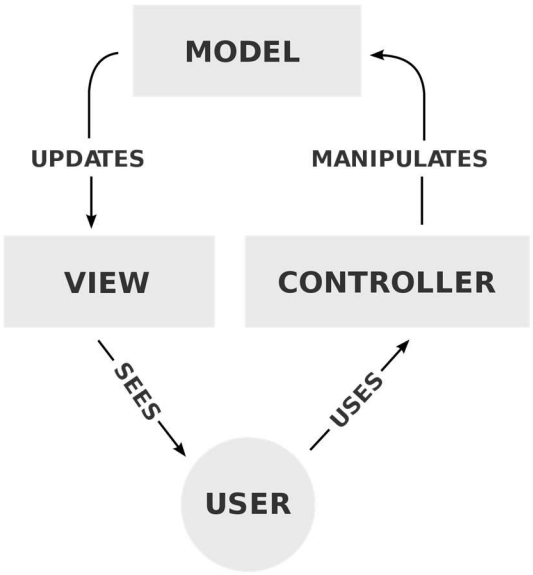


Figure 18: MVC Software Design Pattern

The primary benefit of using the MVC pattern is that it separates the parts of the system into clear reusable components. The backend for a web application or mobile application may require the same functionality but different user interfaces. Separating out the functionality in this way allows us to simply swap the view and controller for different platforms. That is, the application is not tied to any one platform, a concept which has become increasingly important as the types of devices we use to run software becomes more diverse. For example, desktop, web, smart phone, smart watch, smart speaker and so on.

Most modern programming languages will implement the MVC pattern when developing software that has a user interface. Consider the programs that you have created in Visual Studio, a form window can be considered a view and the controller is the code that is attached to that form. We can create other C# classes which implement the functionality of a program and allow us to interact with data.

Building a Database

Today's exercises are adapted from the Microsoft C# Visual Studio tutorials. We are going to create a simple application which will "mock" a database setup. Typically, in order to use a

database an application would need to connect to a server. That server would then receive requests to modify the database. Consider the following image.

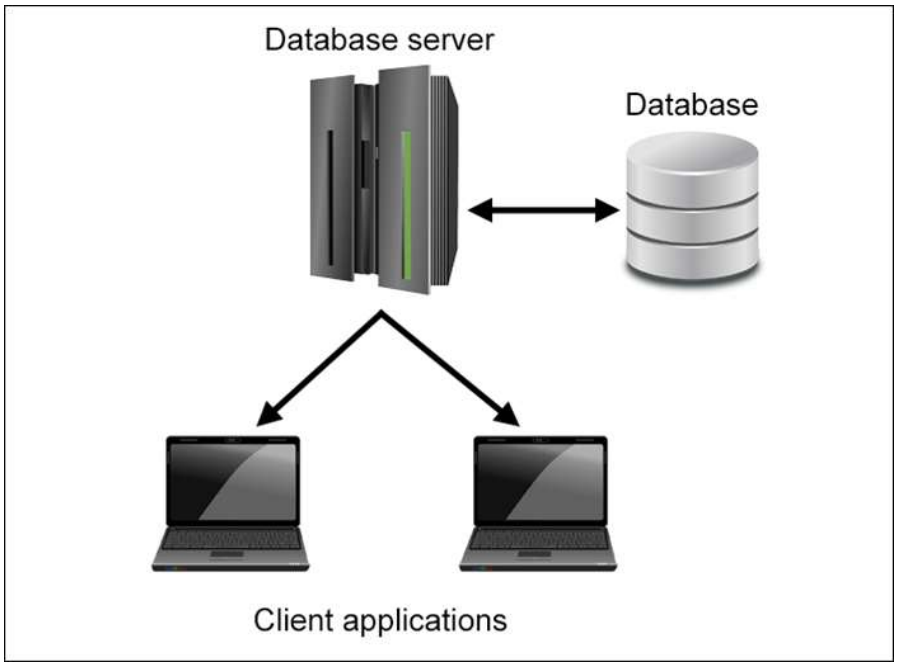


Figure 19: Database Architecture

A piece of software runs on several different computers. That software connects to the database server where the database is stored. The server maintains the connection between the software and the data, including ensuring that safe and secure connections are established. The database can only be modified via the server, that is, the client application cannot modify the database directly.

In Visual Studio, the Integrated Development Environment (IDE) allows us to use the computer we are running the software on to also be a "local server". This essentially allows us to mock the typical database architecture by using the machine we are running our software on as a "fake" external server for our application. Therefore, all the software can be developed in a local environment without the cost of using an external server.

Building a Database Application

Before you continue working on your Parkway Commute Application, let's demonstrate how to work with a local database in Visual Studio. Open Visual Studio and create a new Windows Form App (.NET Framework) for the C# programming language. Name the project "DatabaseExample".

Setup the Local Data Server

1. Open the SQL Server Object Explorer window.

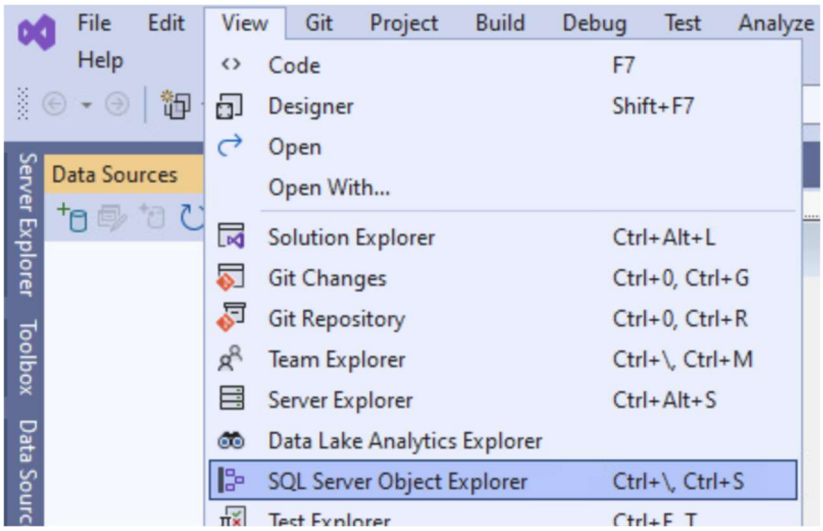


Figure 20: SQL Server Object Explorer Menu

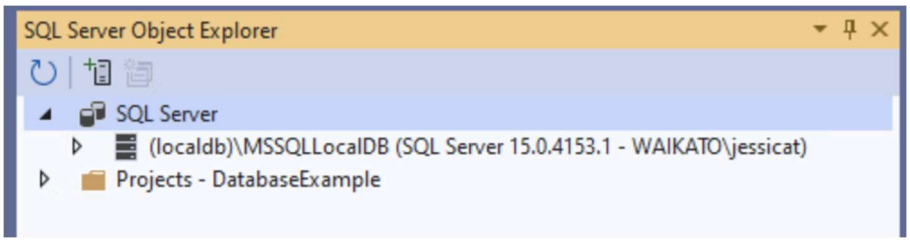


Figure 21: SQL Server Object Explorer

2. Create a New Query by right clicking on the local database instance (localdb).

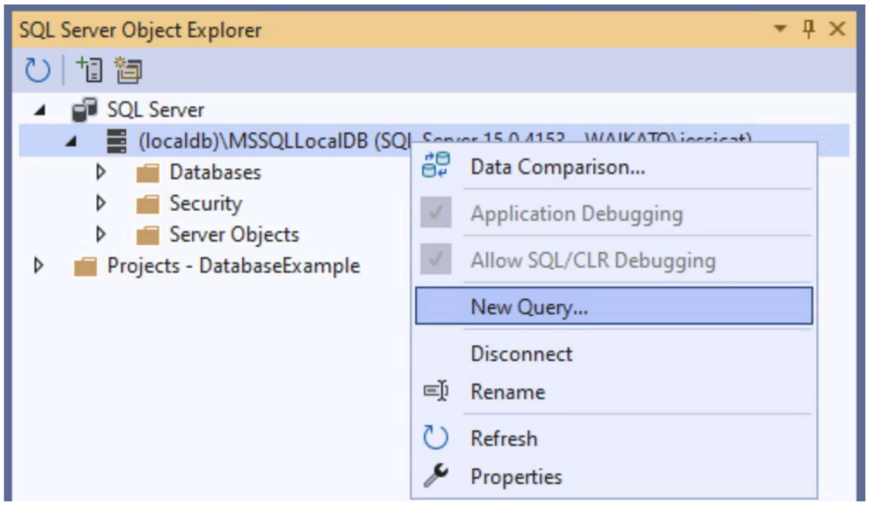


Figure 22: Create a New Query

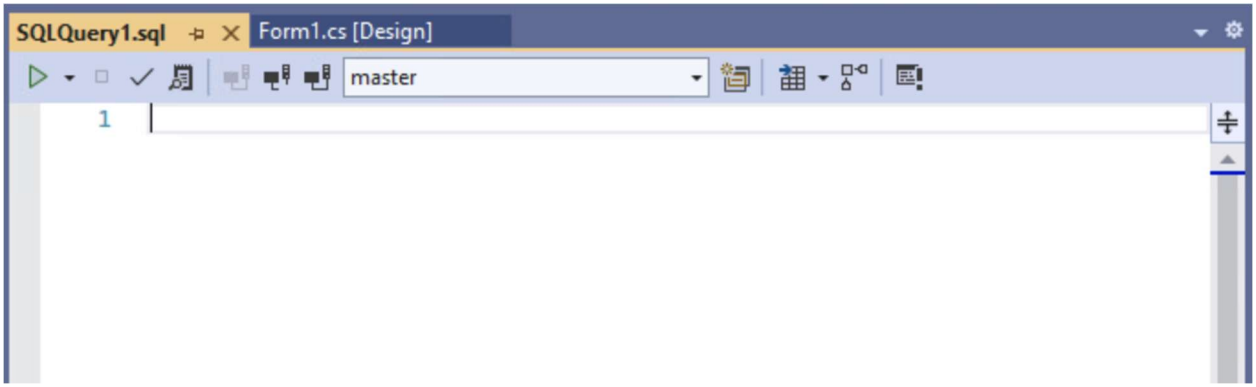


Figure 23: Query Window

3. Copy the Northwind Transact-SQL script[1] and paste it into the query window.

Figure 24: Pasted Northwind Script

4. Select the execute button.

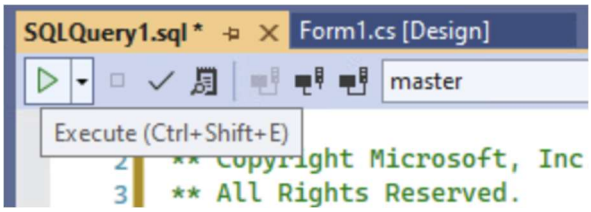
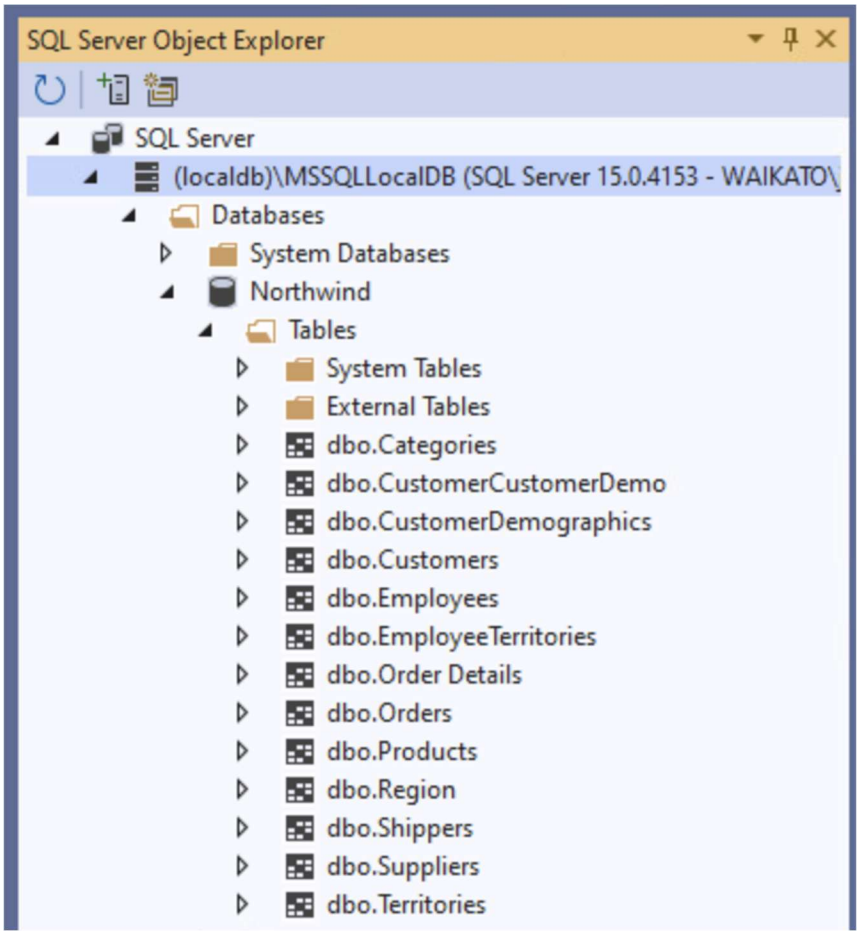


Figure 25: Execute Button

If the query executed successfully you should see the following output.

[1] https://raw.githubusercontent.com/MicrosoftDocs/visualstudio-docs/main/docs/data-tools/samples/northwind.sql

Figure 26: Commands Completed



Figures 27: Database Added Successfully

5. Be sure to save or copy your server name (e.g. `(localdb)\MSSQLLocalDB`) as you will need this in the next exercise.

Create the Data Source

1. Open the data sources window.

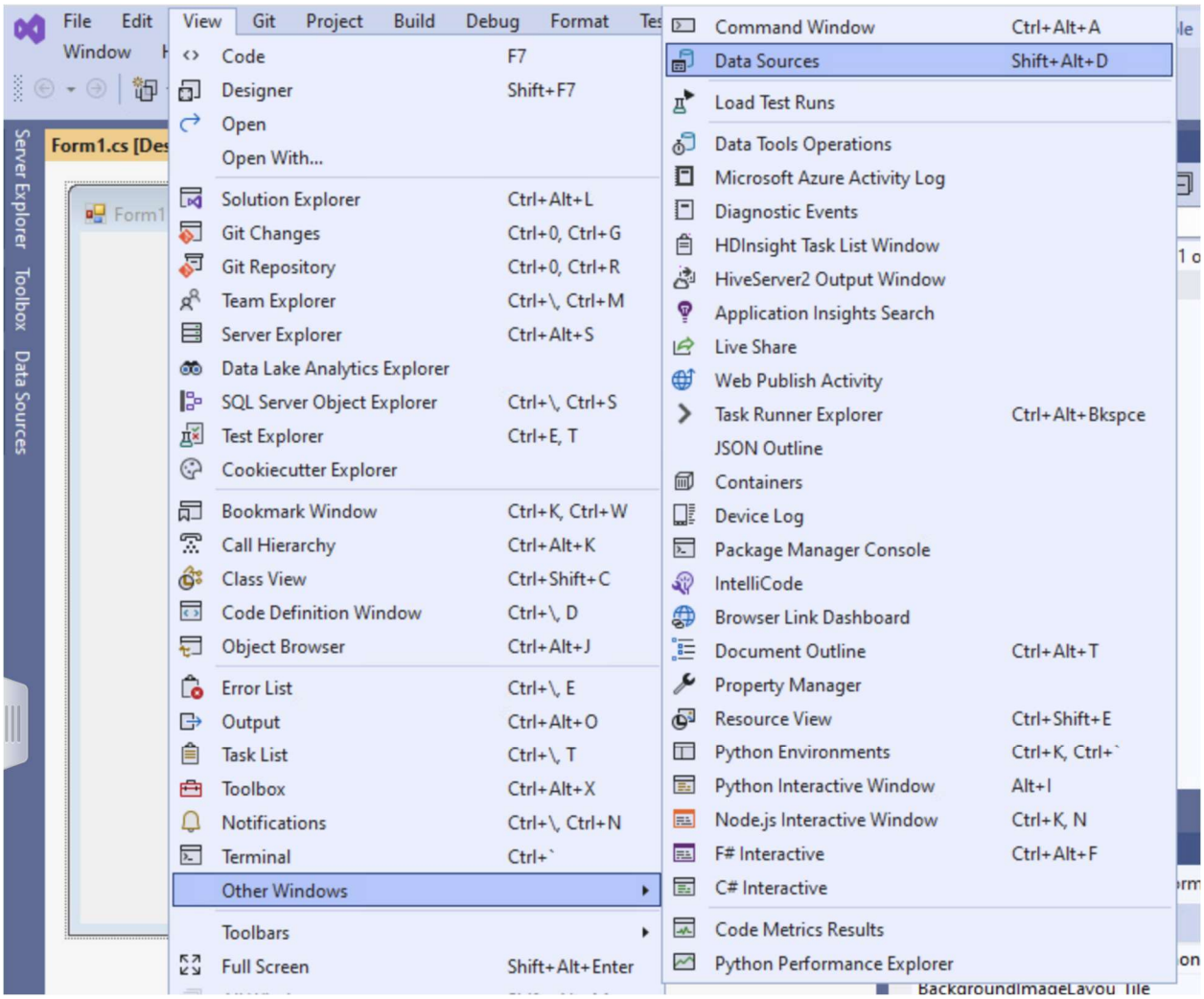


Figure 20: Open Data Sources Window

When you open the window be sure to have your project selected as shown below, otherwise you will have a disabled data sources window.

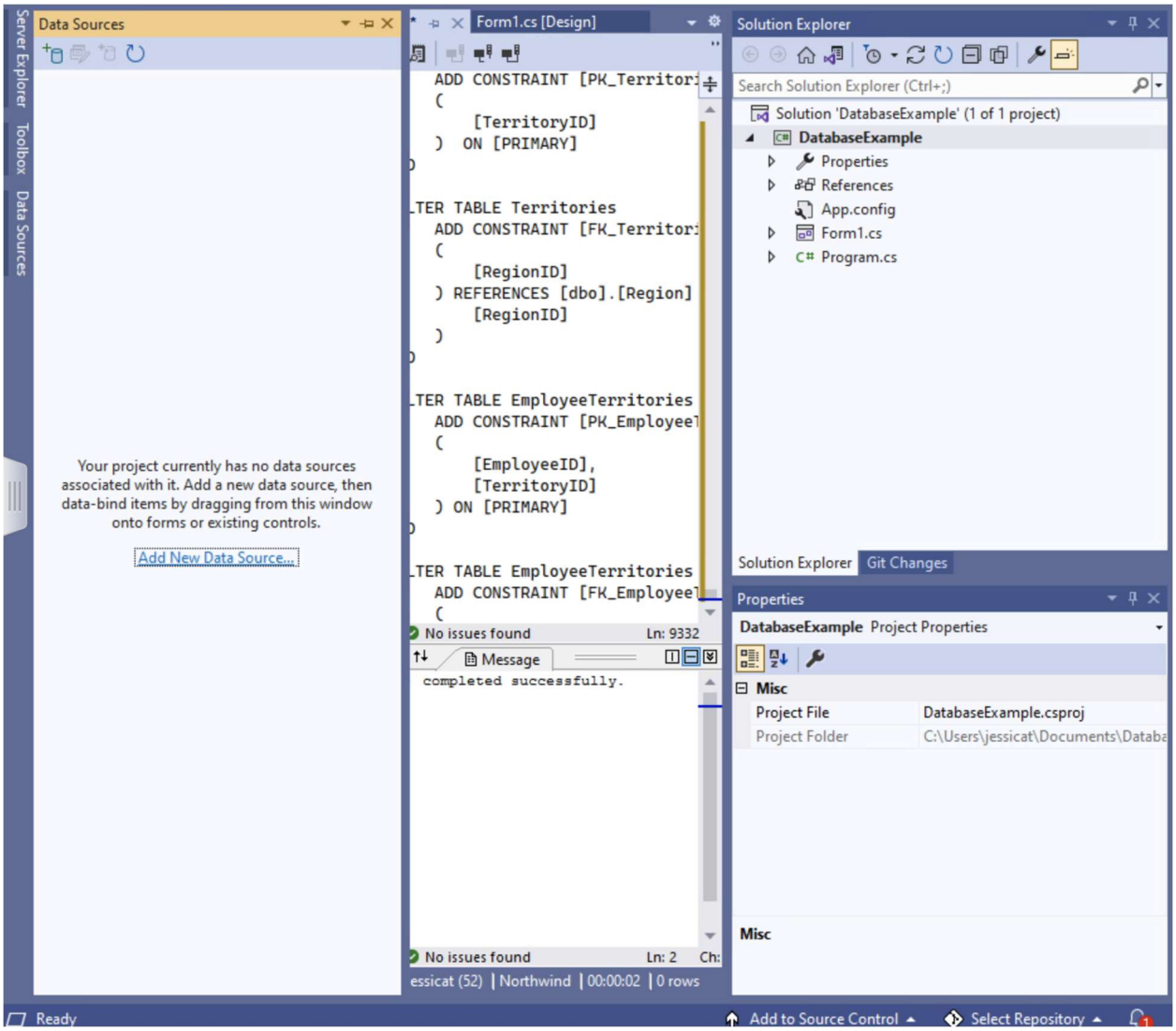


Figure 21: Data Sources Window

2. Select the "Add New Data Source" button.

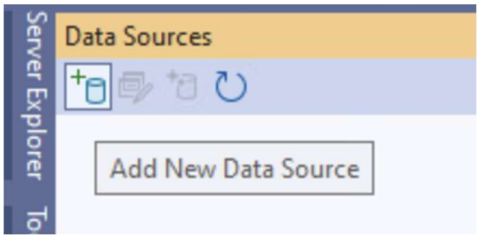


Figure 22: Add New Data Source

3. When the prompt opens select "Database" option and click the next button.

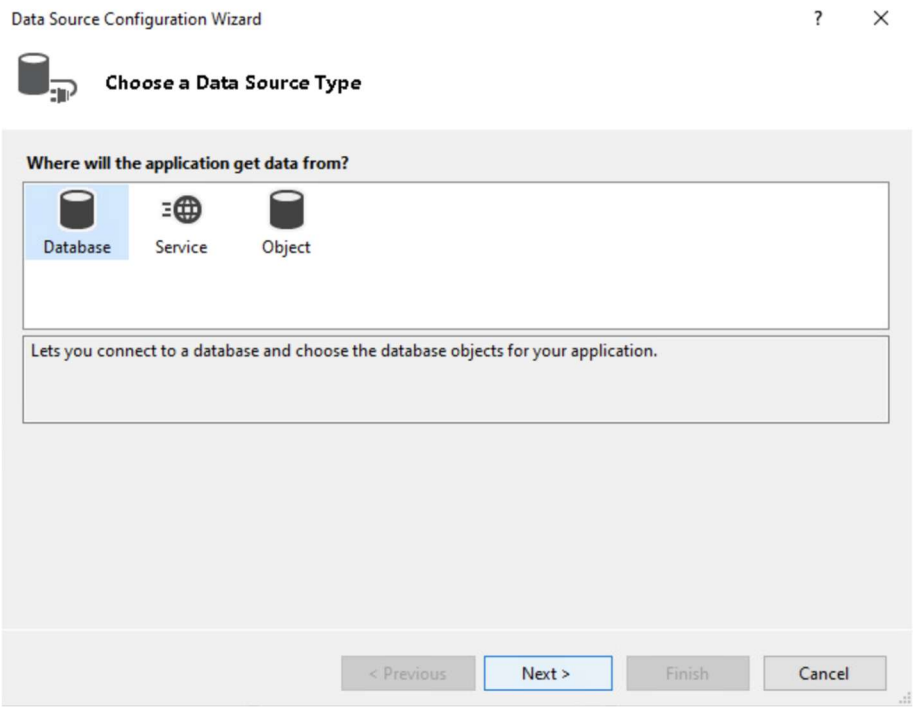


Figure 23: Choose a Data Source Type

4. Select the "Dataset" option and click the next button.

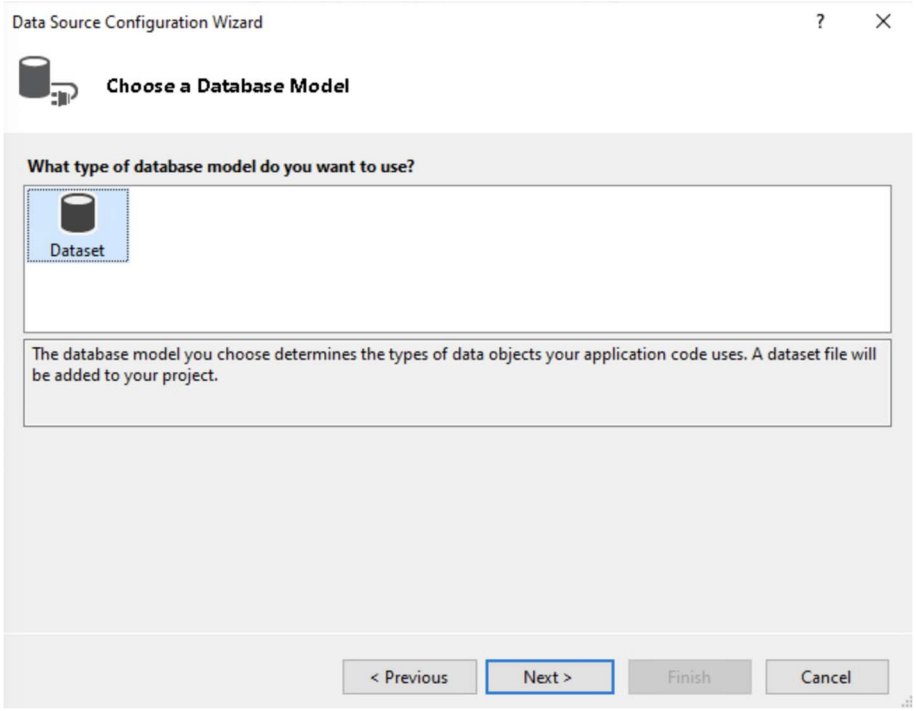


Figure 24: Choose a Database Model

5. Select the “New Connection” button. On the window select the “Microsoft SQL Server” option and click continue.

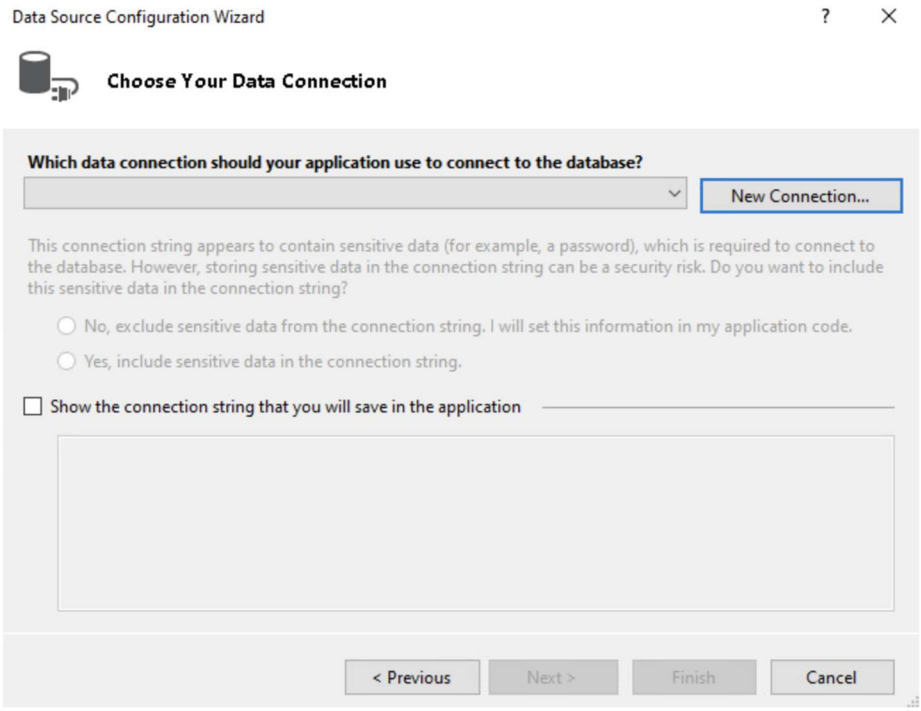


Figure 25: Choose Your Data Connection

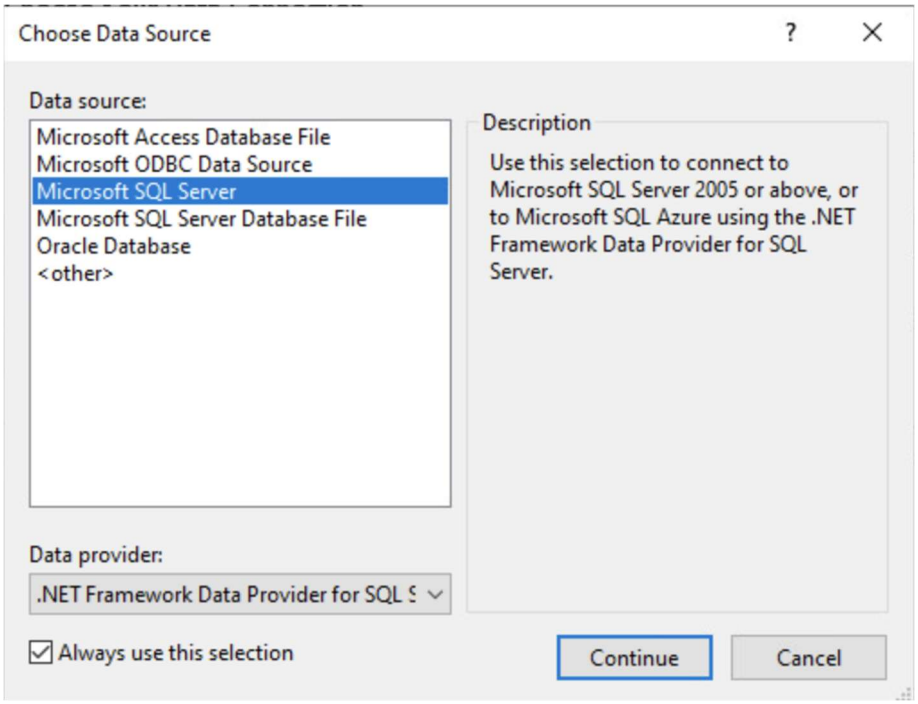


Figure 26: Choose Data Source

6. Type the name of your local database instance into the Server name field and select the Northwind database. Then click the OK button.

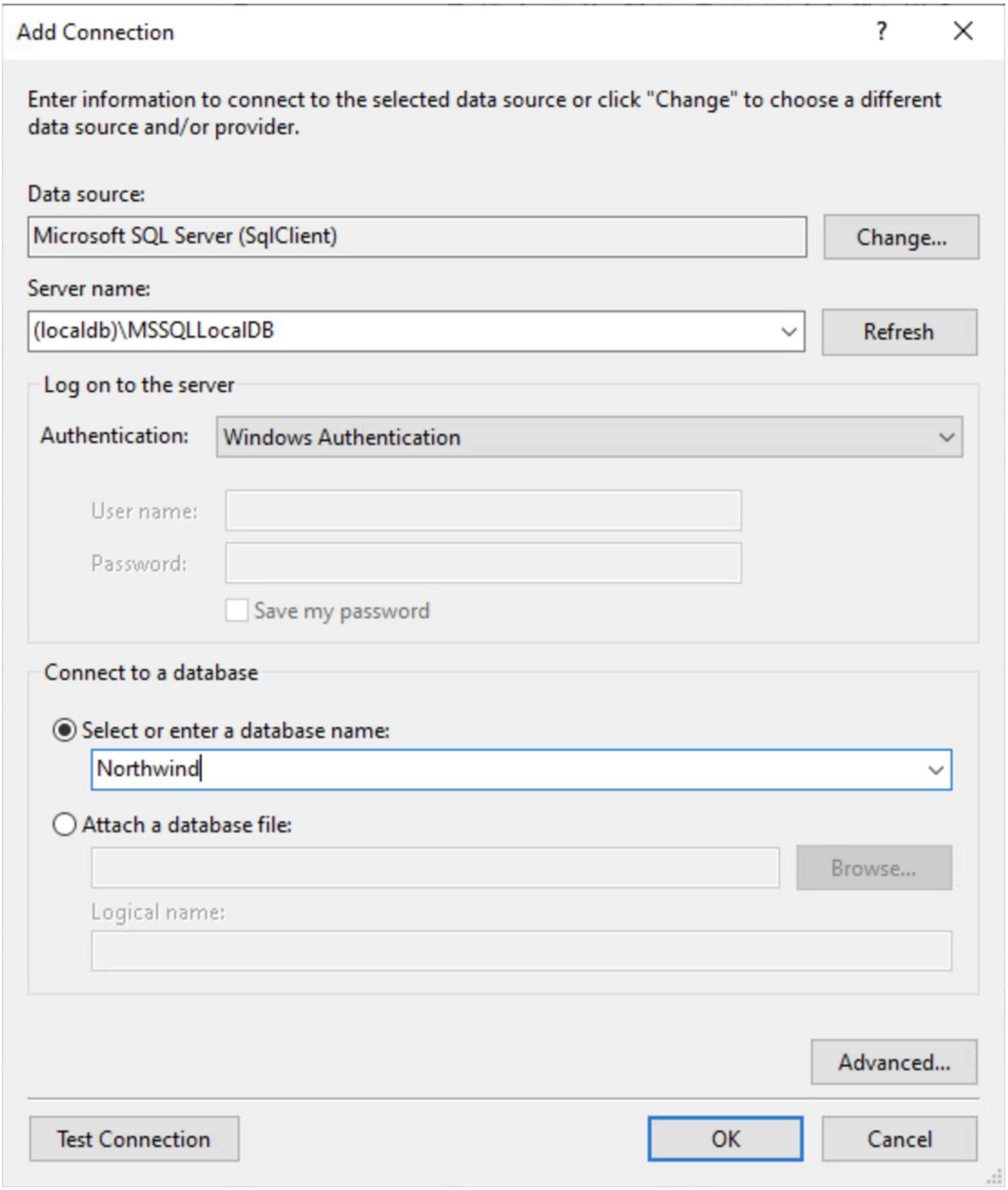


Figure 27: Add Connection

7. The connection will now be loaded, click the next button on the remaining window.

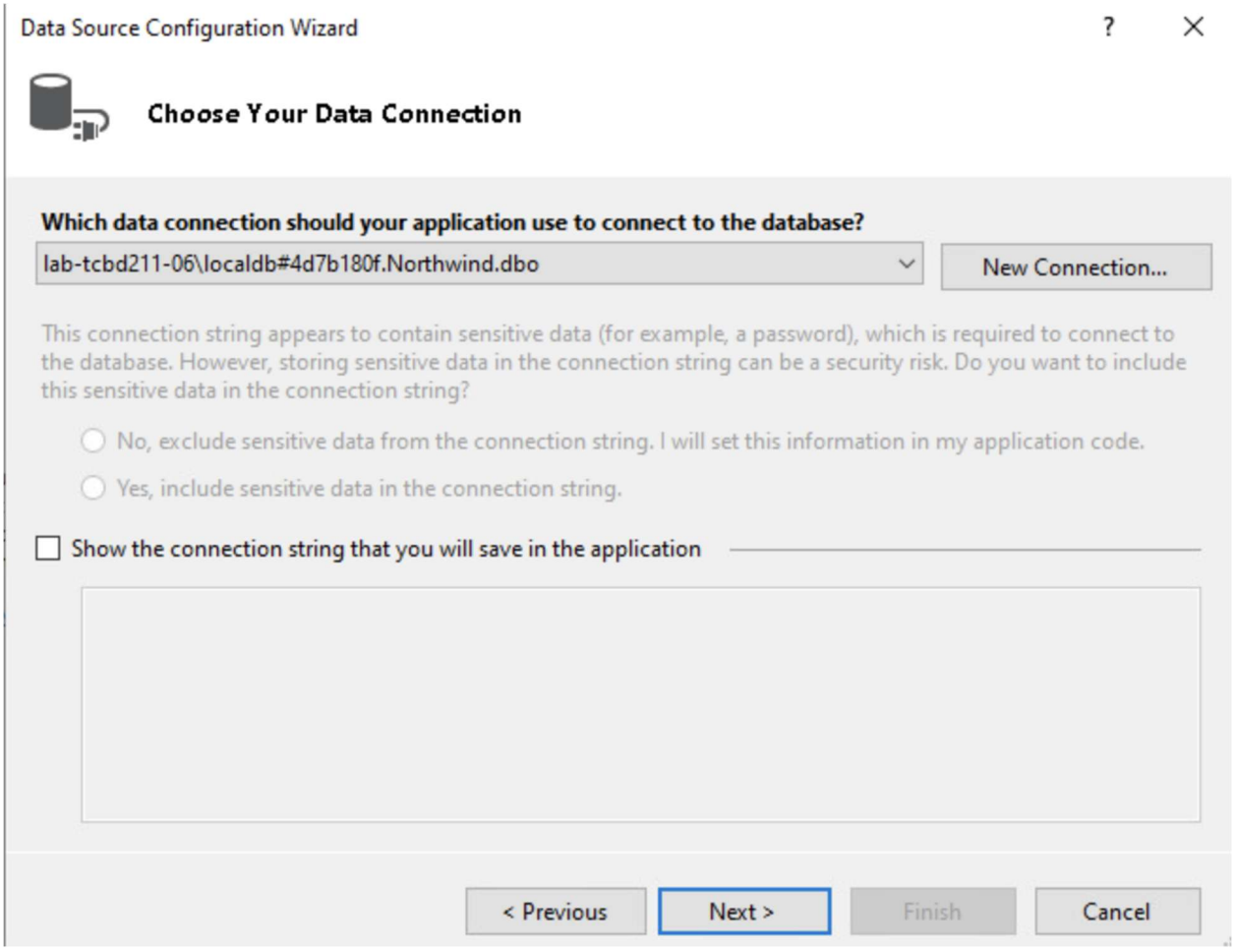


Figure 28: Choose Your Data Connection with String

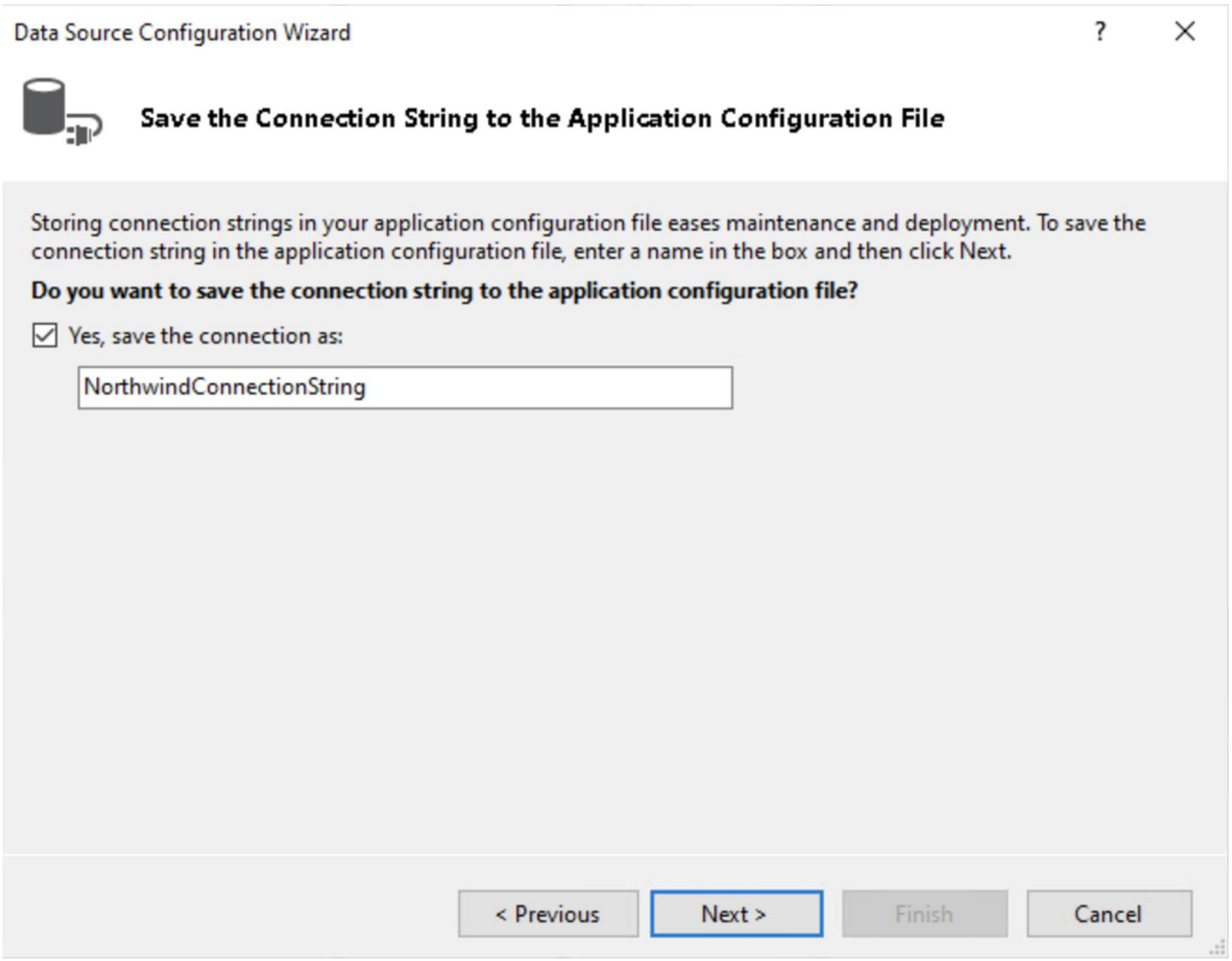


Figure 29: Save the Connection String

8. On the next window, select the tables option and click the finish button.

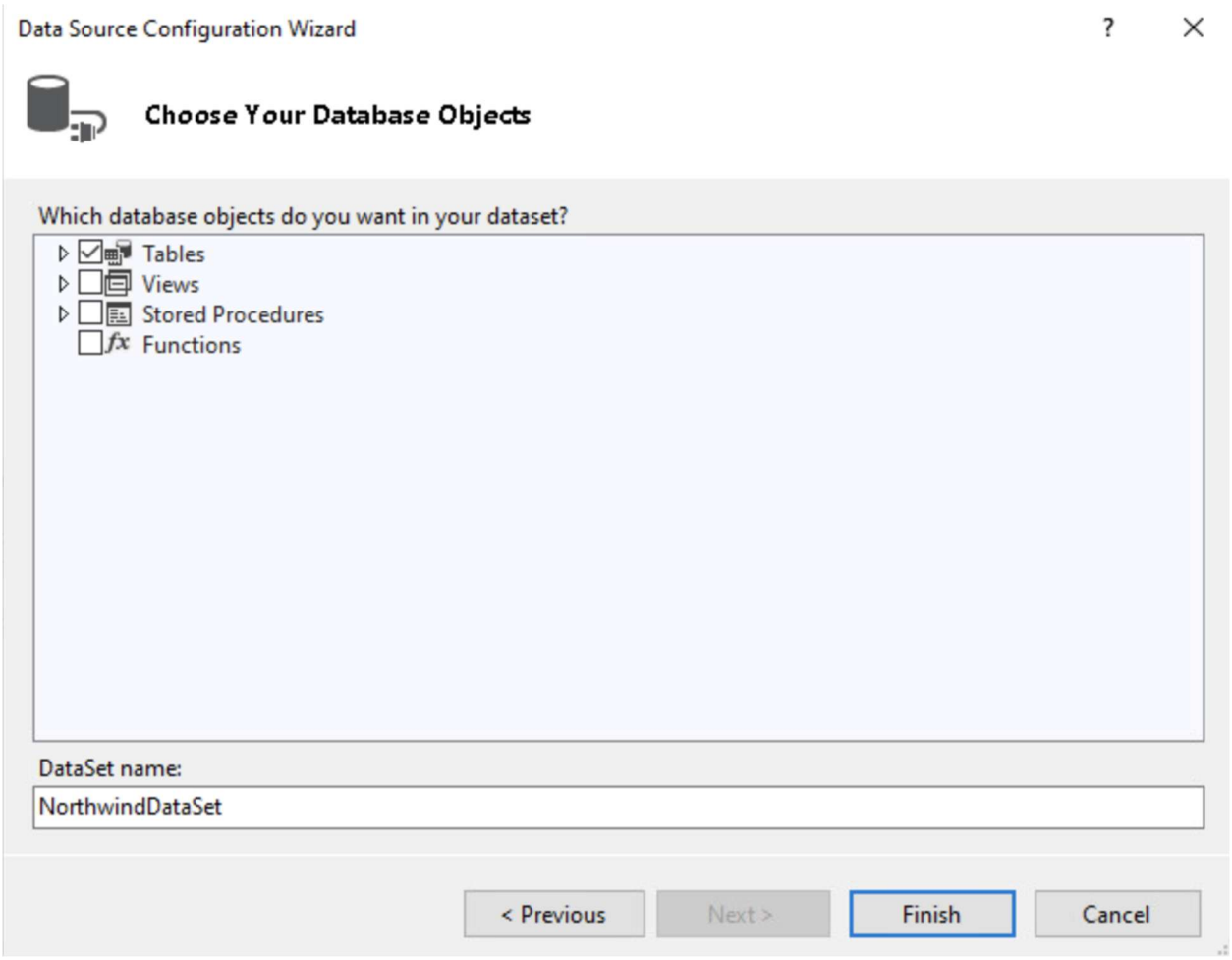


Figure 30: Choose Your Database Objects

If you have completed this successfully you should see the following in your solution explorer.

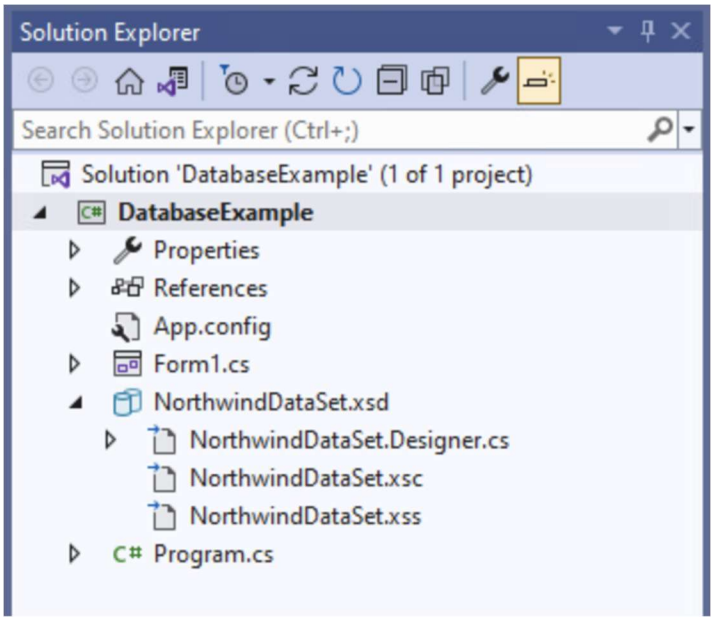


Figure 31: Database Added to Solution Explorer

View the Data on the Form

1. Make sure your windows form is open and the data sources window is visible. Expand the NorthwindDataset to locate the customers table. Drag and drop the customers table onto your form.

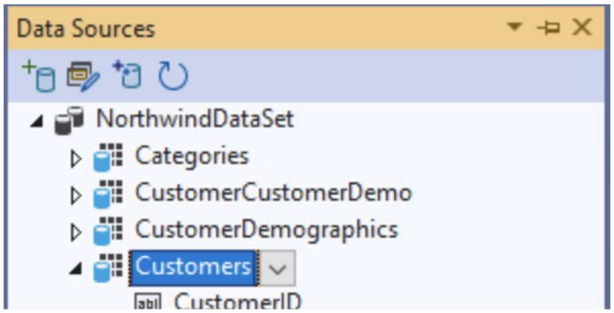


Figure 32: Data Sources

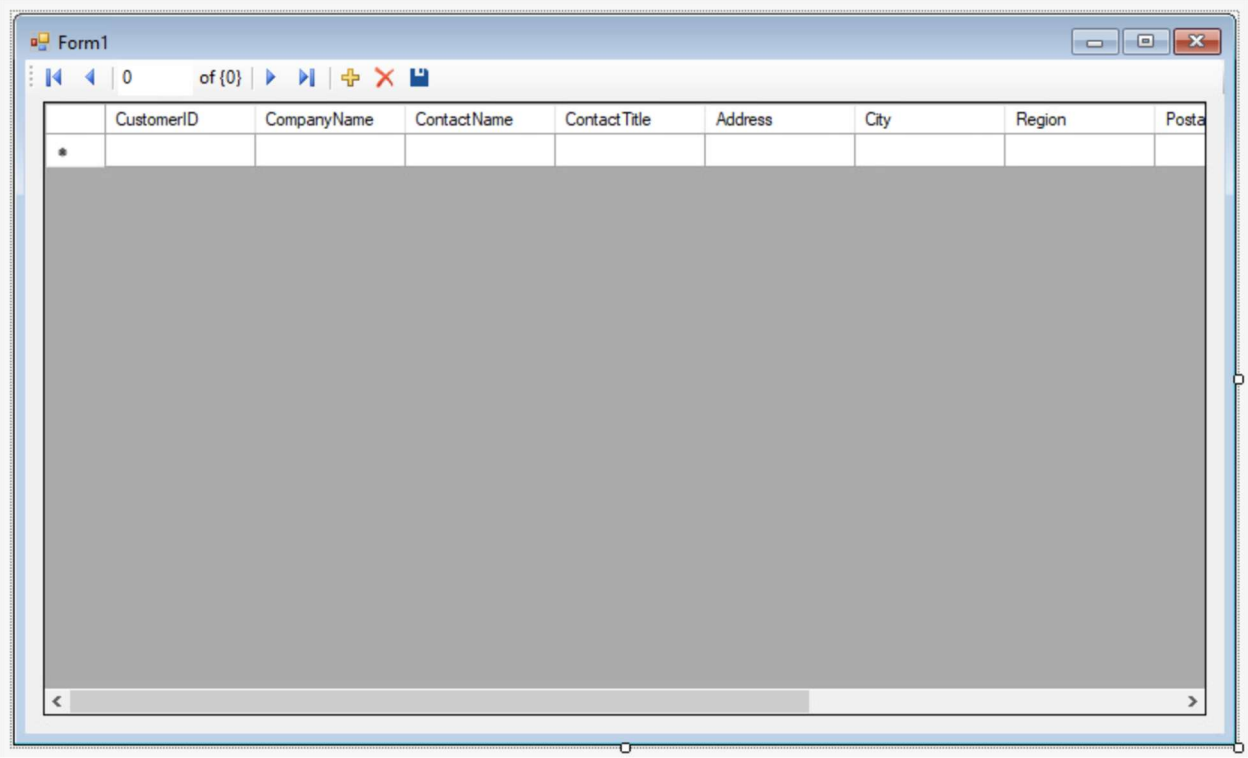


Figure 33: Form1 After Adding Customers Data

You should notice that the form populates automatically with different widgets. This will allow the user to explore the dataset. Try running the application and seeing what happens when you explore the data. Can you add a new customer?

2. Below the form design select the `customersTableAdapter`. Then select the "Add Query" option in the properties window.

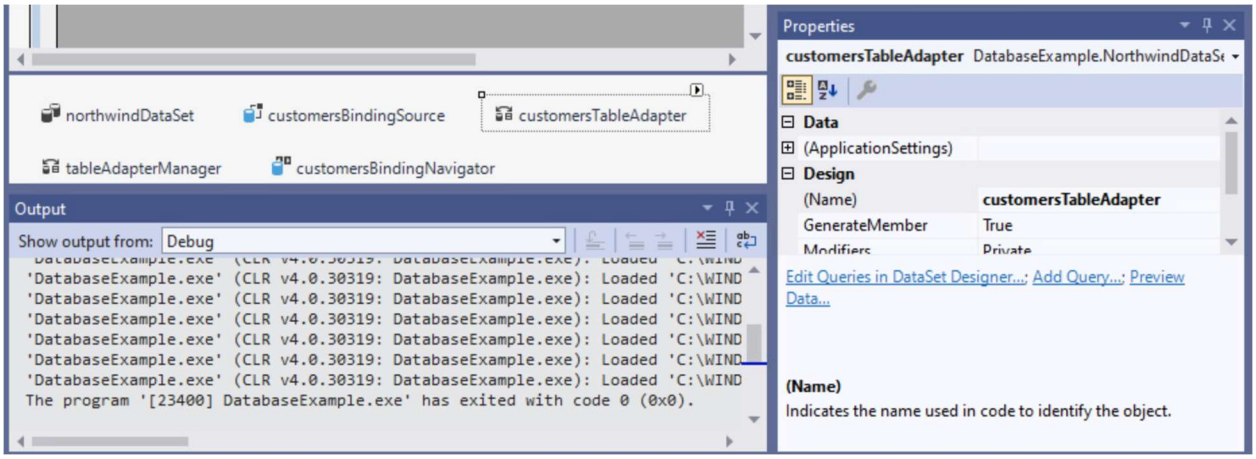


Figure 34: Adapter and Add Query

3. In the Search Criteria Builder give your new query the name "FillByCity" and add the following query text:

```
SELECT CustomerID, CompanyName, ContactName, ContactTitle,
Address, City, Region, PostalCode, Country, Phone, Fax
FROM dbo.Customers
WHERE City = @City
```

Then click the OK button. You should notice the fillByCityToolStrip appear on the tool bar.

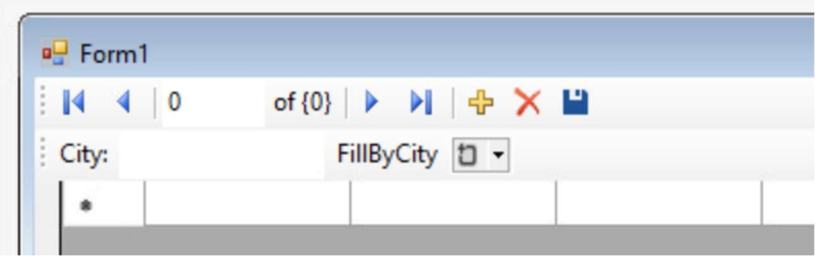


Figure 35: Fill by City Tool Strip Widget

Try running the program again. How does the tool strip modify the data in the window? Can you filter by different cities? How do you refresh the data?

Today's Exercise

Now that you have some experience with adding data to a form we have supplied a SQL script (`ParkwayCommute.sql`) with generated data from Parkway Commute. Modify your application that you started on last week to add the data and view it. If you are having any issues with adding information then ask a staff member for assistance. They will be able to help you construct the necessary SQL queries in order to get the correct data displayed.

Once the data has been added and can be viewed, review your requirements and use cases. Which of these can you implement using this process?

For an advanced exercise, try creating a database to store usernames and passwords following the format in the `ParkwayCommute.sql` script. How would you ensure that this worked?

Summary

Today we have covered some of the basics of backend development, specifically looking at the Model-View-Controller design pattern. By the end of today's session you should have a project which has a user interface and connection to a database. Next session we will start exploring

more complex functionalities that you could add to your application to implement more of the initial project requirements. Be sure to save your application in a safe spot (e.g. Google Drive) so that you can open it again next time.

Useful Resources

- MVC Magic!: https://avelonpang.medium.com/mvc-magic-13e37d782cf7
- Create a Windows Form to search data: https://docs.microsoft.com/en-us/visualstudio/data-tools/create-a-windows-form-to-search-data?view=vs-2022
- SQL queries syntax: https://www.w3schools.com/sql/sql_syntax.asp