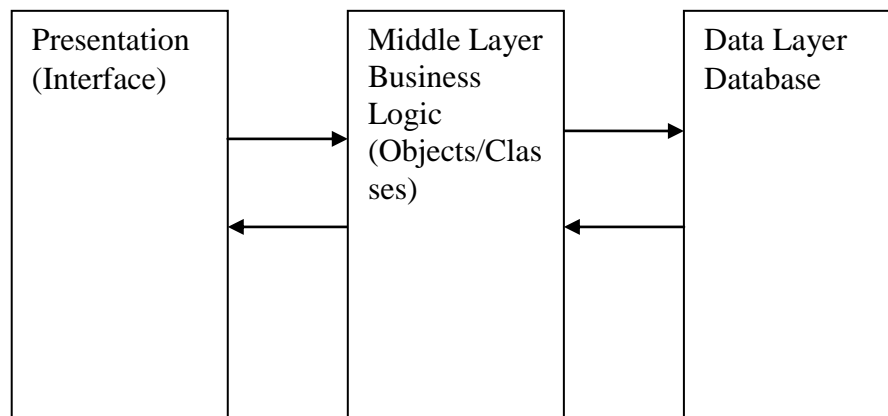


Creating the Data Layer

When interacting with any system it is always useful if it remembers all the settings and changes between visits.

For example, Facebook has the details of your login and any conversations with friends. If we use the system and then log off and return to it the next day it remembers everything that we did the day before. This is achieved by storing the data in a database.

In terms of the architecture we are looking at for this module the database is stored inside the data layer.



The data layer of a system is where the data is stored in some sort of database. Inside the database are typically the following.

- Tables – where we organise the data for long term storage and retrieval
- Stored Procedures – written using SQL allowing us to manipulate the data in the tables

There are several database systems available for example

- Access
- MySQL
- SQL Server
- Oracle

For this module, we are going to use SQL server from within Visual Studio.

Guided Tour - Overview of the Database Structure

For the first part of this work I don't want you to write any code but rather take a bit of time looking at a completed version of the system you will be creating.

Follow the instructions below and just have a go at looking around an existing program.

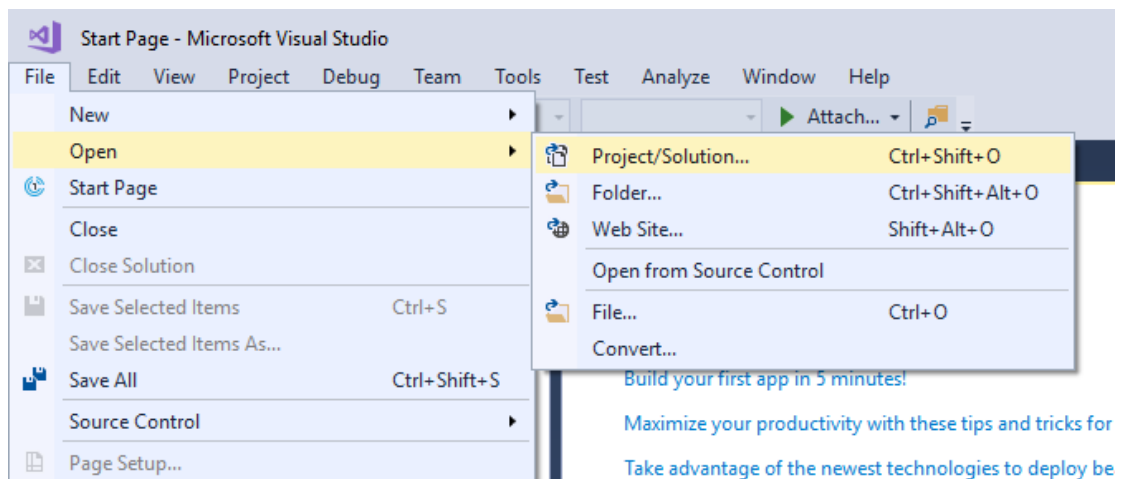
From the module web site download the zip file for Address Book Finished.

IMPORTANT – make sure that you have extracted the zip file. If in doubt ask your tutor.

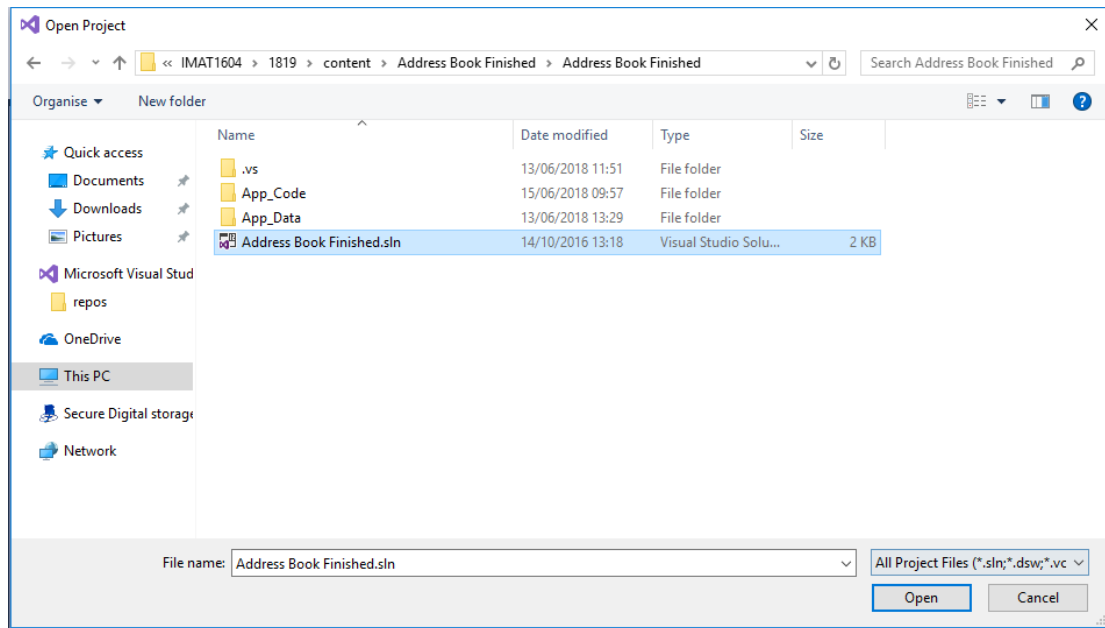
The next step is to open Visual Studio.

You should be able to start Visual Studio from the Start Menu.

To open an existing web site, select file – open – Project/Solution

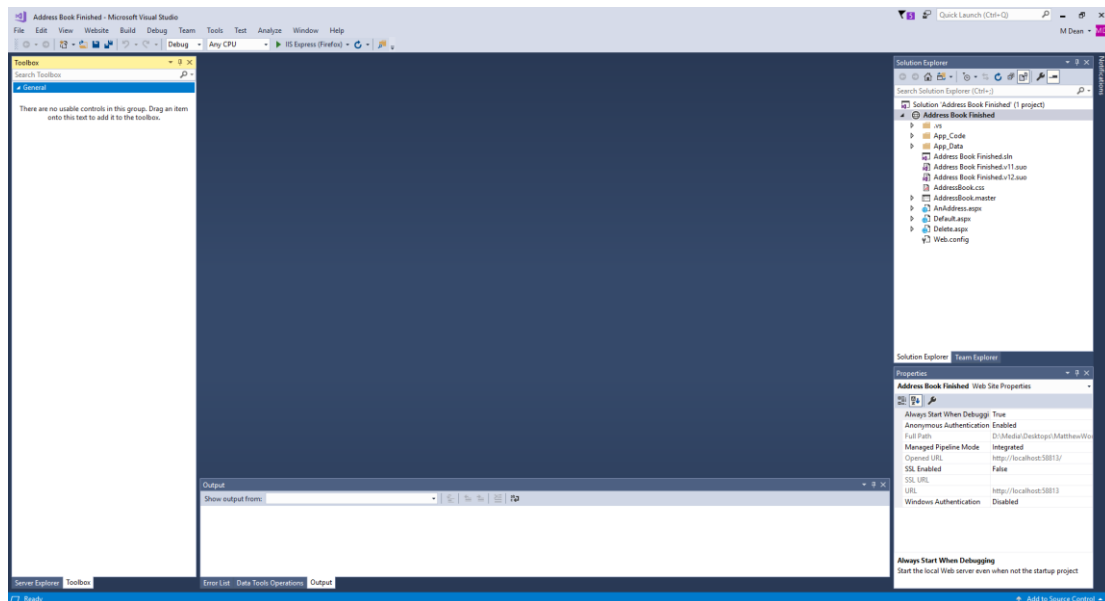


You should see the following screen where you need to navigate to the folder where you extracted the above file...

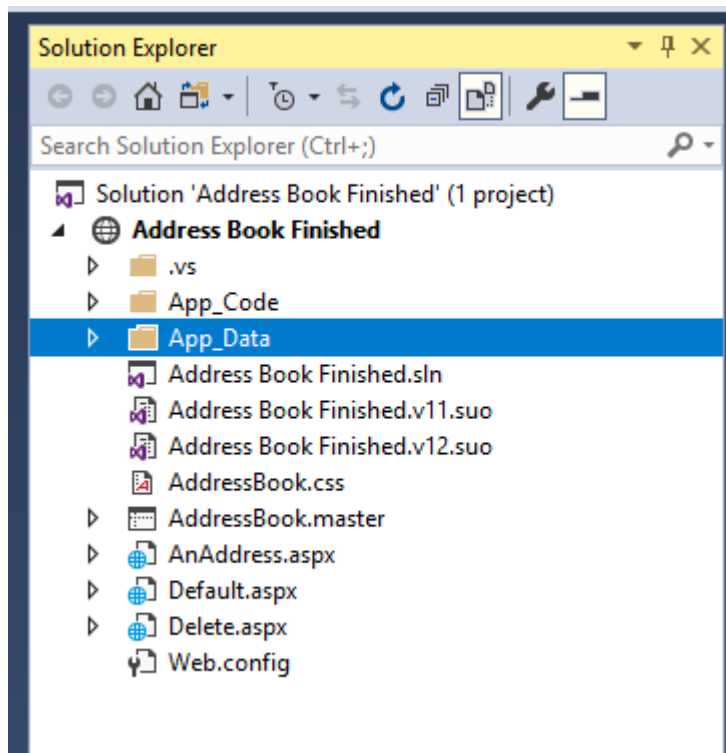


You will know that you have the correct folder as you can see the App_Data and App_Code folders. Click on the Address Book Finished folder and press open.

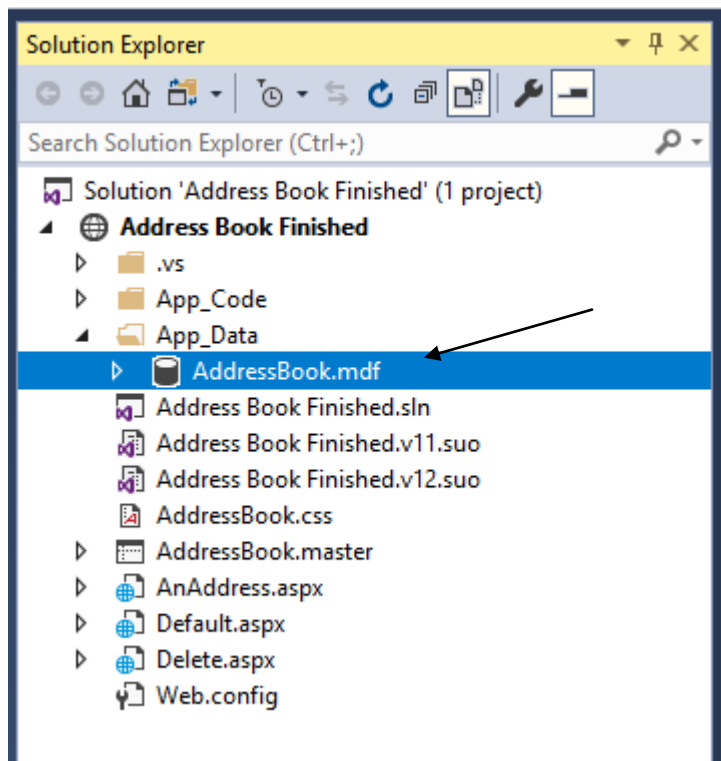
You should see the following...



To the right of the interface is the solution explorer. This is where you will see all the files for your program...

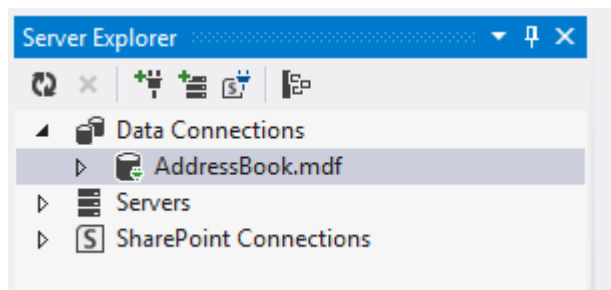


Notice the folder called App_Data which contains the database for this system. Expand the folder by clicking the small triangle and you will see the database file...



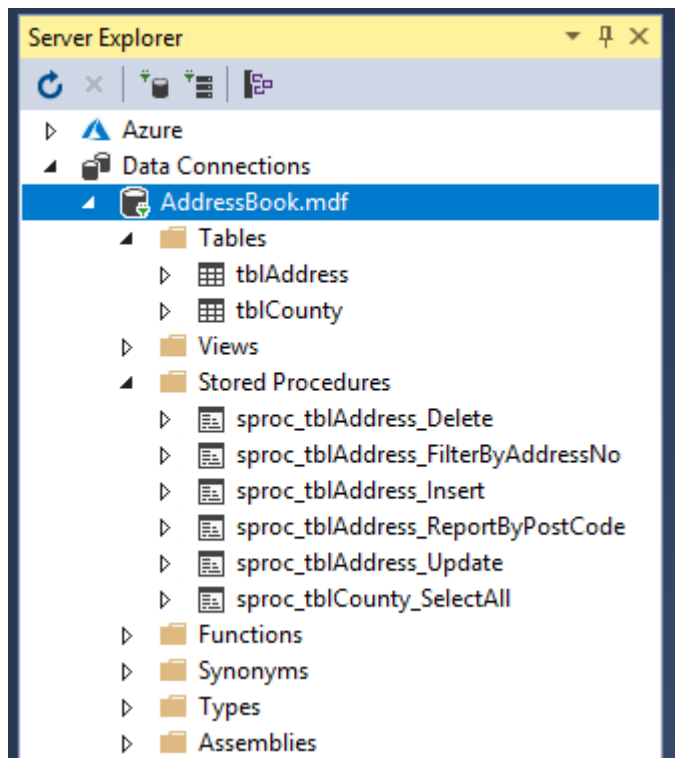
Double click the database file to open it in Visual Studio.

To the left of the program is the server explorer...



This allows you to edit the database file for this program.

Click on the triangle to the left of the database name to expand the database structure...



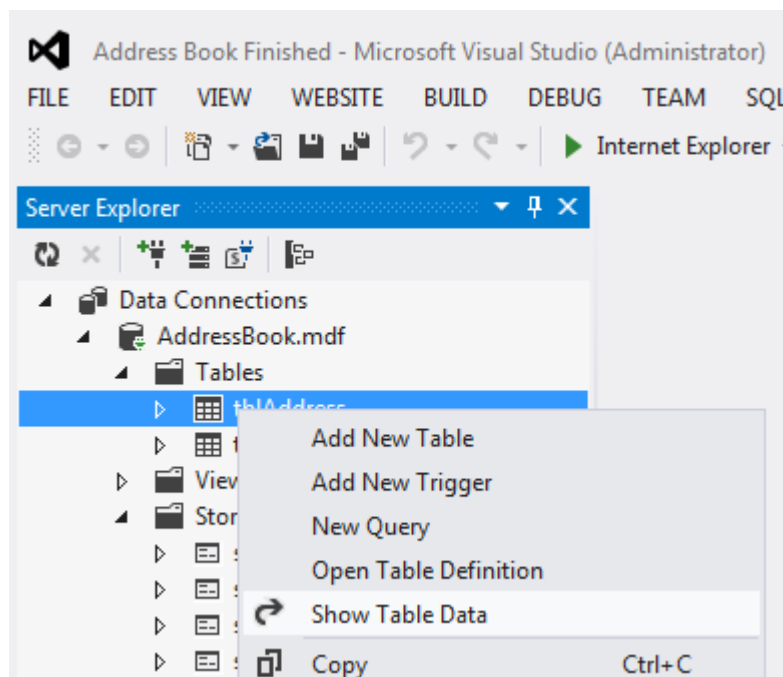
In this module, we are not going into any depth on database design but we will cover enough to get started. We will be creating some tables and some stored procedures.

Notice how the different entities in the database are named. Tables are being given the prefix tbl, stored procedures spproc. Naming conventions are a way of naming objects so that they are easier for the programmer to identify.

Notice also the two types of item in this database...

- Tables give us a place to organise and store the data
- Stored procedures allow us to manipulate the data in the tables

Expand the tables section and right click on the table tblAddress selecting show table data...



You should see the data stored in the table...

	AddressNo	HouseNo	Street	Town	PostCode	CountyCode	DateAdded	Active
▶	2	1	Some Street	Leicester	LE1 1BE	35	07/09/2012	True
	3	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
	4	33	High Street	Leicester	LE1 6FG	35	07/08/2012	True
	5	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

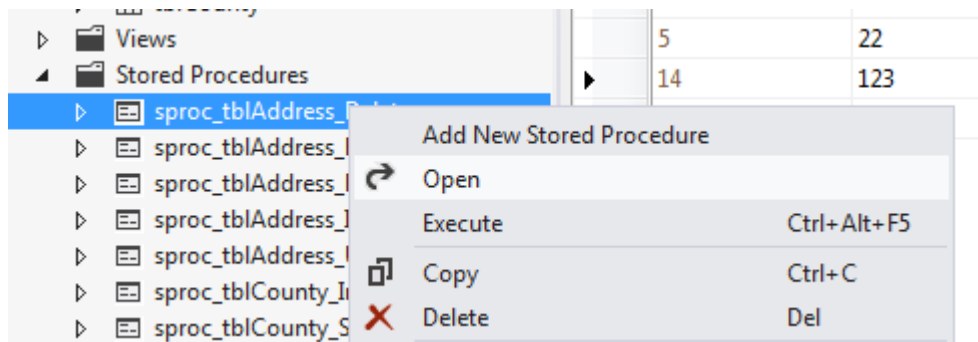
Try adding an extra record. (You will find that you will not be able to type into the AddressNo and Active fields.)

	AddressNo	HouseNo	Street	Town	PostCode	CountyCode	DateAdded	Active
	2	1	Some Street	Leicester	LE1 1BE	35	07/09/2012	True
	3	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
	4	33	High Street	Leicester	LE1 6FG	35	07/08/2012	True
	5	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
▶	14	123	Another Stree	Leicester	LE3 1EE	35	20/05/2013	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

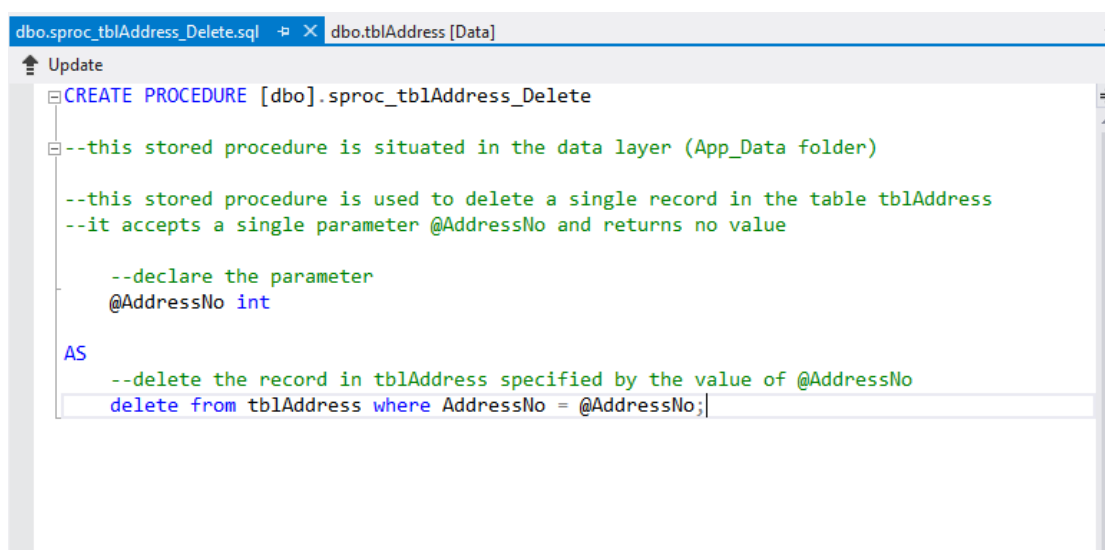
Notice that the AddressNo value was generated automatically for you. This field is called the primary key and is used to uniquely identify each entry in the table.

Make a note of the primary key for the record you just created (in this case 14).

Close the table and now open the stored procedure called sproc_tblAddress_Delete...



This will display the code for the stored procedure...



Don't worry about what this all means yet, just get an idea of where things are.

Stored procedures are a tool in the database systems that allow us to manipulate the data in the table.

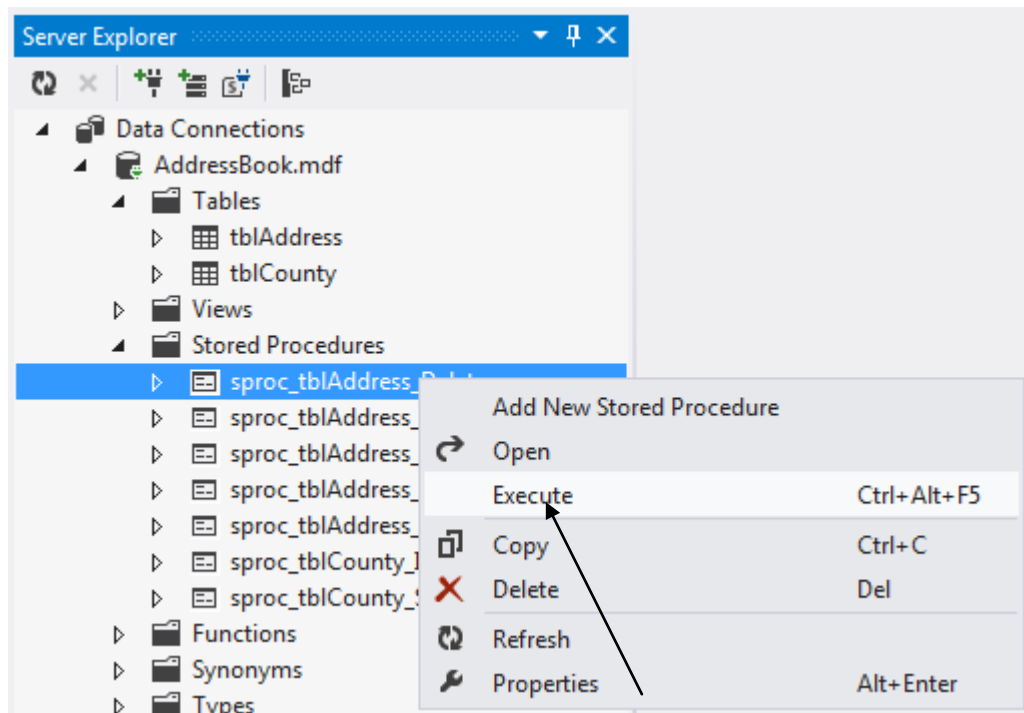
There are typically four main things we want to do with the data in a database...

- Obtain a list of records that may or may not be filtered in some way
- Add a new record to the table
- Update an existing record in the table with new data
- Delete a record from the table

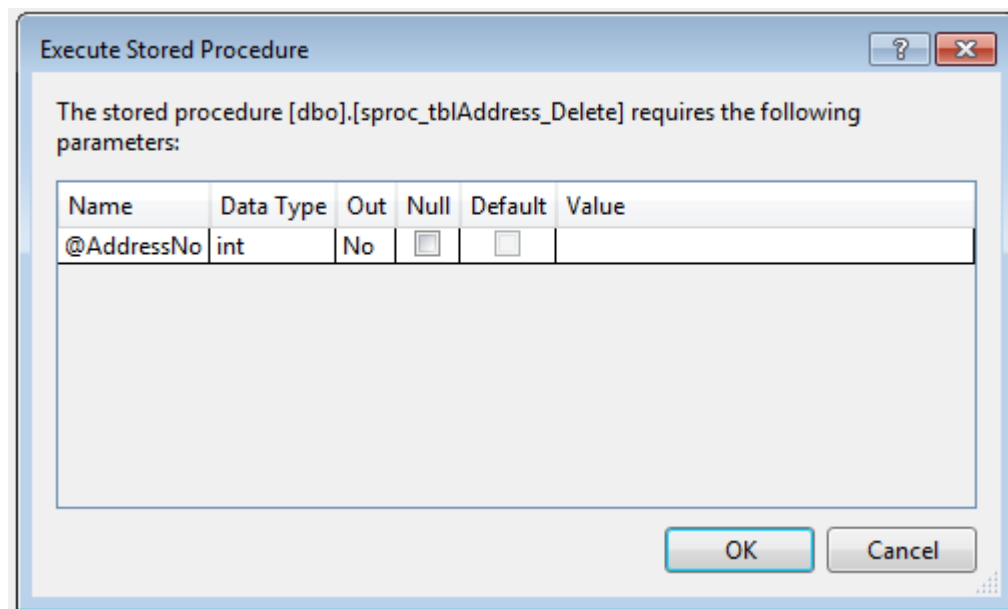
Stored procedures are written in a database query language called Structured Query Language or SQL.

In this example, the stored procedure will accept a single parameter, the primary key value of a record we want to delete.

Close the stored procedure and now execute it like so...

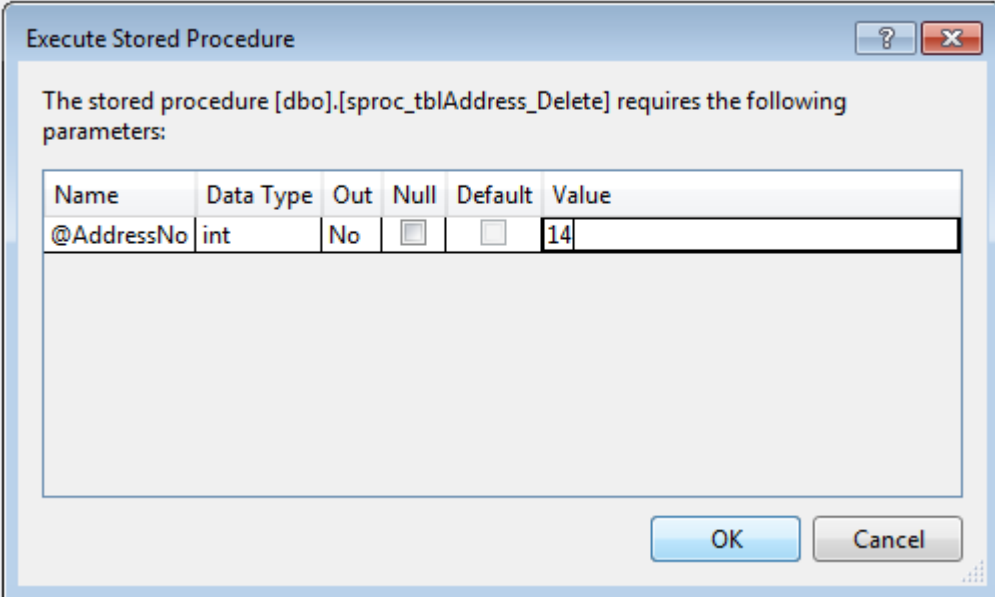


You should see the following screen...



Here the stored procedure is asking for the unique identifier value of the record to be deleted.

Enter the number of the record to delete (in the example above 14 but yours might be different).



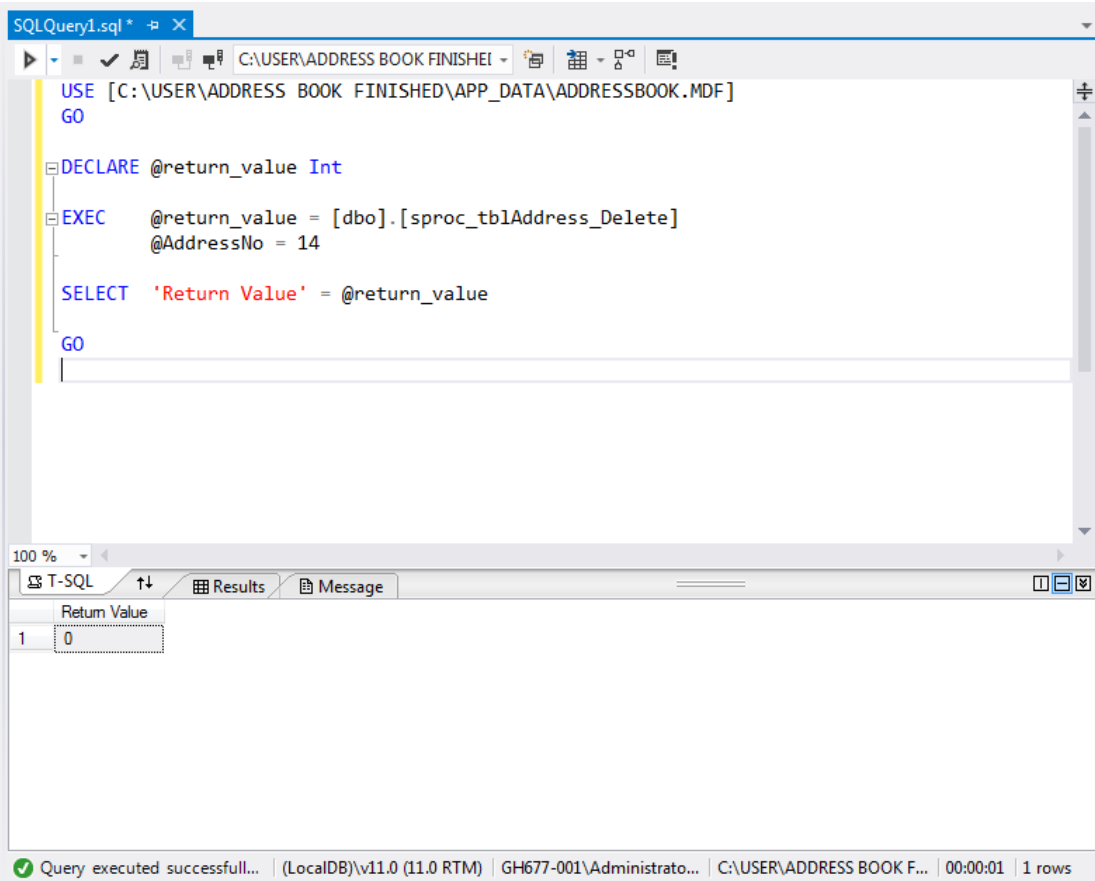
The stored procedure [dbo].[sproc_tblAddress_Delete] requires the following parameters:

Name	Data Type	Out	Null	Default	Value
@AddressNo	int	No	<input type="checkbox"/>	<input type="checkbox"/>	14

OK Cancel

Now press OK.

You should see a screen reporting what happened...



SQLQuery1.sql * X

USE [C:\USER\ADDRESS BOOK FINISHED\APP_DATA\ADDRESSBOOK.MDF]
GO

```
DECLARE @return_value Int  
EXEC @return_value = [dbo].[sproc_tblAddress_Delete]  
    @AddressNo = 14  
SELECT 'Return Value' = @return_value  
GO
```

100 %

T-SQL Results Message


Return Value
1 0

Query executed successfully... | (LocalDB)\v11.0 (11.0 RTM) | GH677-001\Administrato... | C:\USER\ADDRESS BOOK F... | 00:00:01 | 1 rows

Note at the bottom it should say that the Query executed successfully.

Close this screen and look at the data in the table again...

	AddressNo	HouseNo	Street	Town	PostCode	CountyCode	DateAdded	Acti
▶	2	1	Some Street	Leicester	LE1 1BE	35	07/09/2012	True
	3	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
	4	33	High Street	Leicester	LE1 6FG	35	07/08/2012	True
	5	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

You should find that the record with the primary key value you passed to the query has been deleted. (If it is still there press the refresh button )

Take another look at the SQL that drives the stored procedure...

```
delete from tblAddress where AddressNo = @AddressNo;
```

Notice how the syntax for the parameter is set up.

Firstly, it has a prefix of @ identifying it as a parameter rather than a field

Notice the logical comparison `where AddressNo = @AddressNo;`

This tells the stored procedure to locate any records in the data where the unique identifier value matches the value of the parameter, in the example above 14.

Lastly notice the action we want to perform in the data matching the logical test...

```
delete from tblAddress
```

Now that you have had a bit of a guided tour of a pre-prepared data layer you will now set about creating your own.

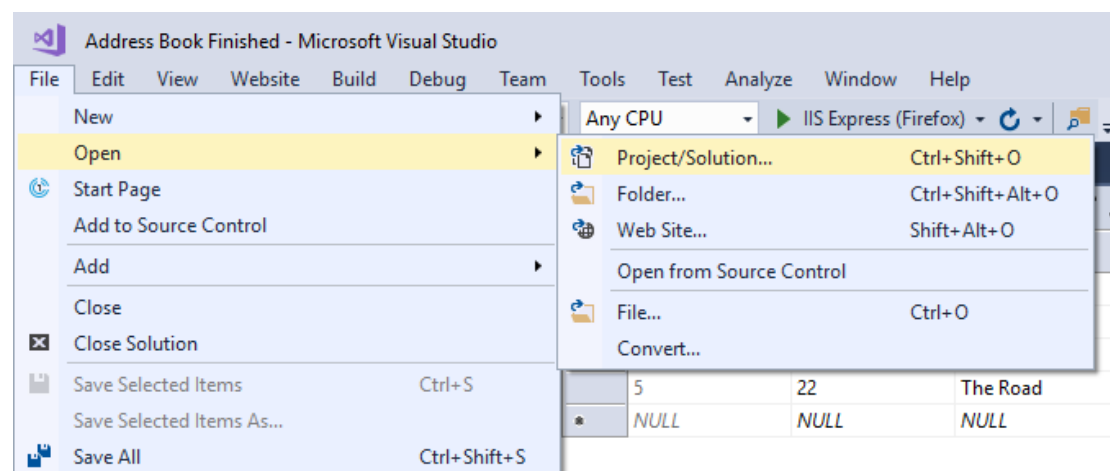
Now Try it yourself

Creating Your Own Data Layer in Visual Studio

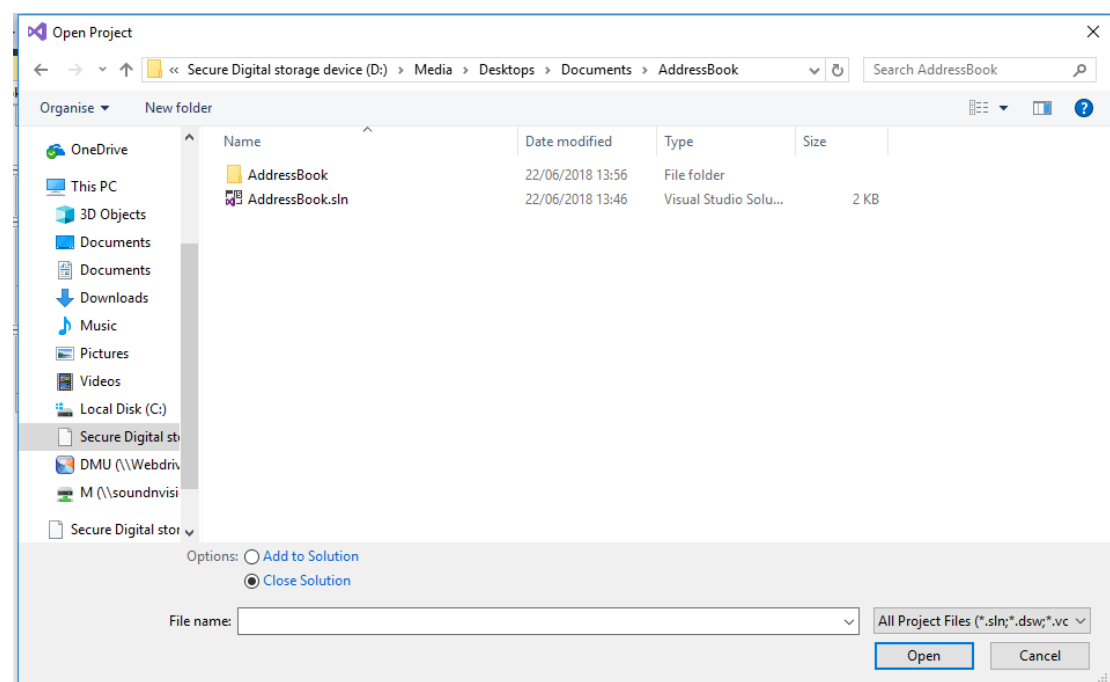
Having run through the guided tour, you can now have a go at building something yourself.

In the last session, you will have created your first web site in Visual Studio and creating your presentation layer.

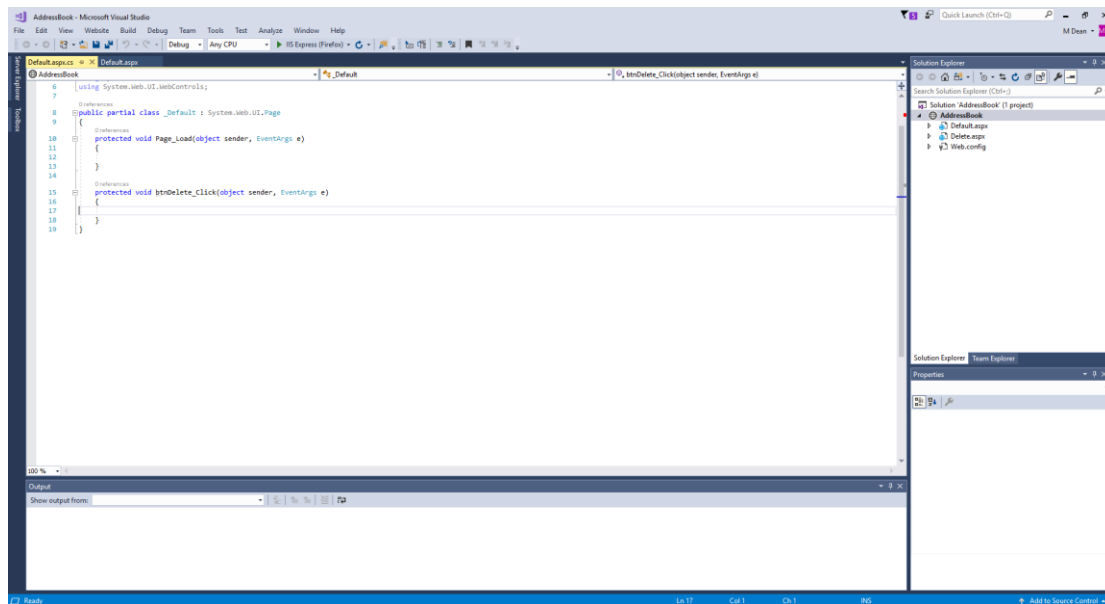
To open your work from last week, start Visual Studio and select Open Project/Solution...



You should navigate to the folder containing your work...

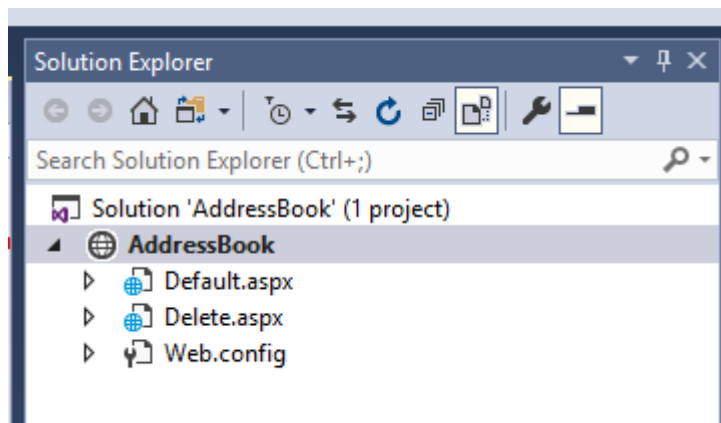


Once you have opened your work from last week Visual Studio should look something like this...

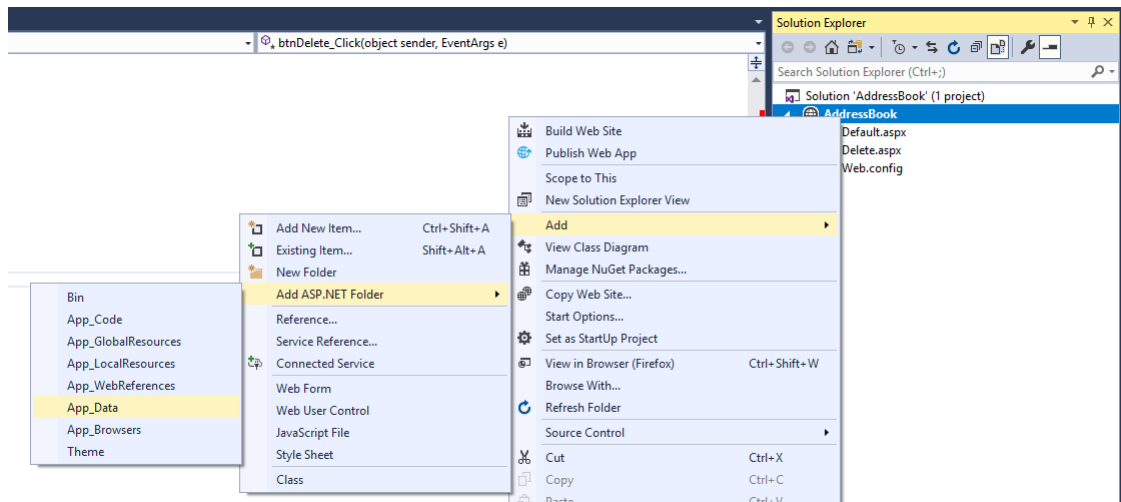


Creating the Data Layer

In the top right of Visual Studio is the Solution Explorer.

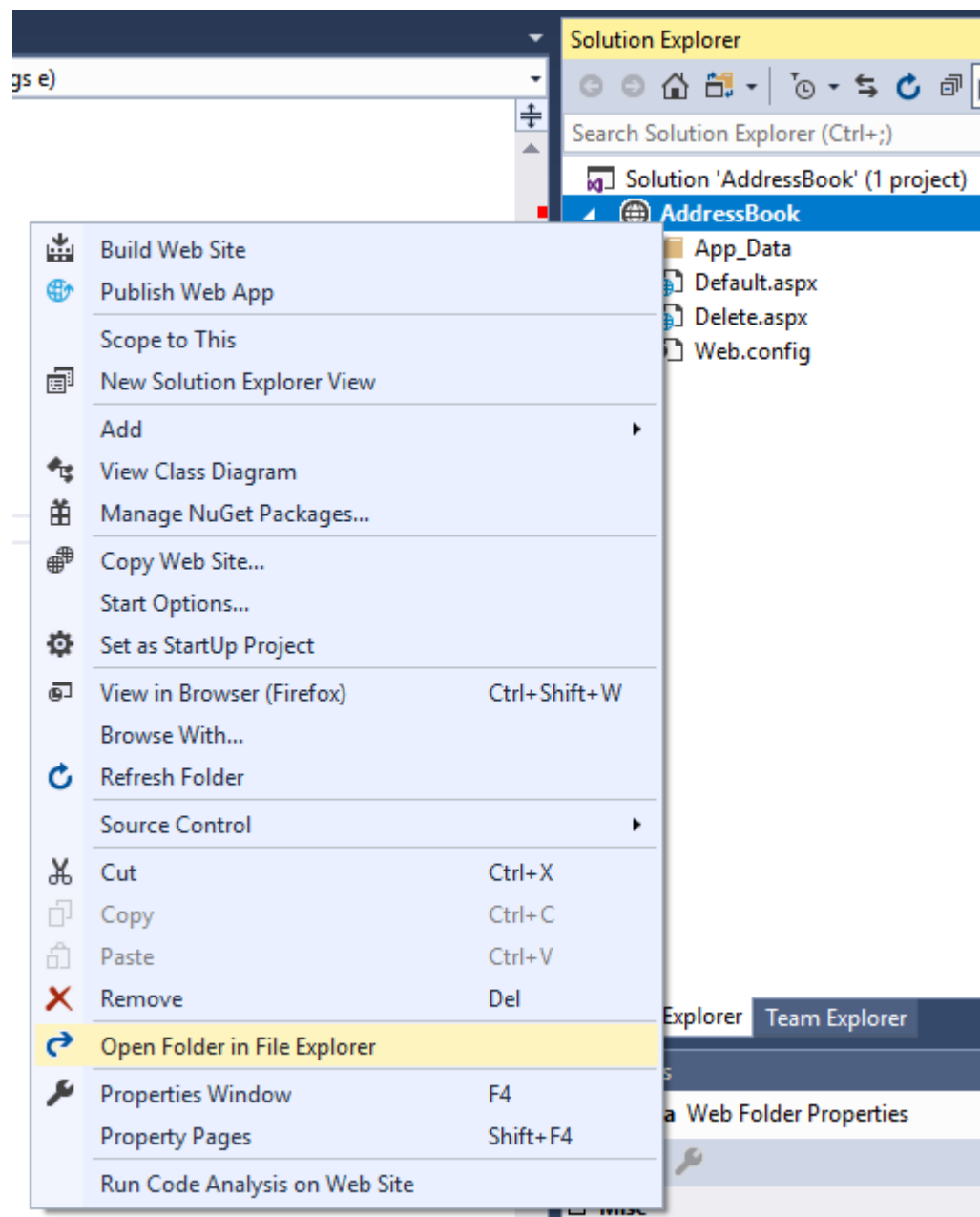


Right click on the web site and add the App_Data folder...

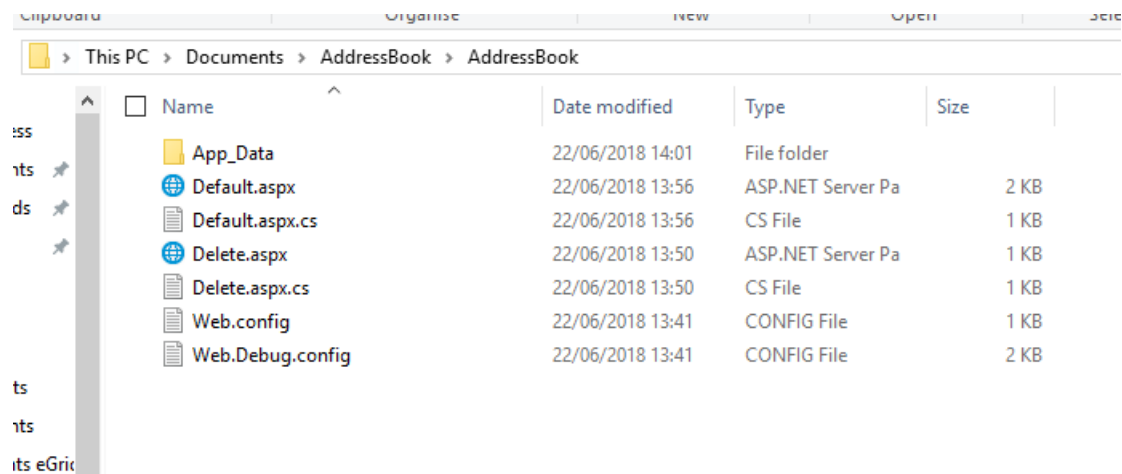


This special folder is where we want to place our database so that we may use it as the data layer in the web site.

Now right click this time on the web site name (AddressBook) and select Open Folder in File Explorer.



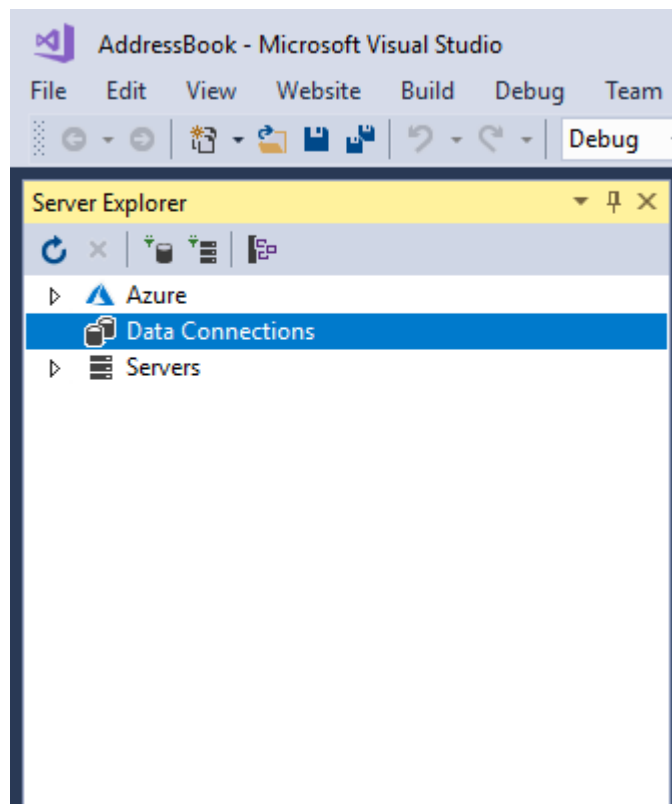
This will take you to where your web site is stored in the file system.



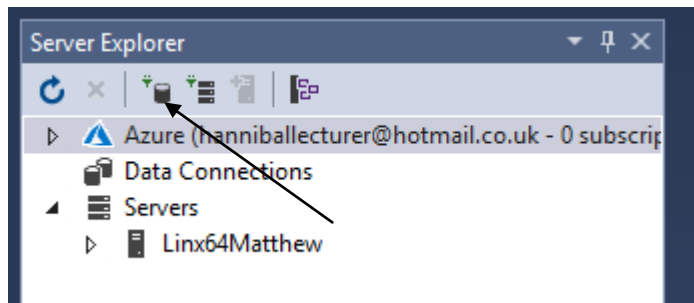
Notice that App_Data is a windows folder set up as a sub folder of the site.

The next step is to create the database file for use in the program.

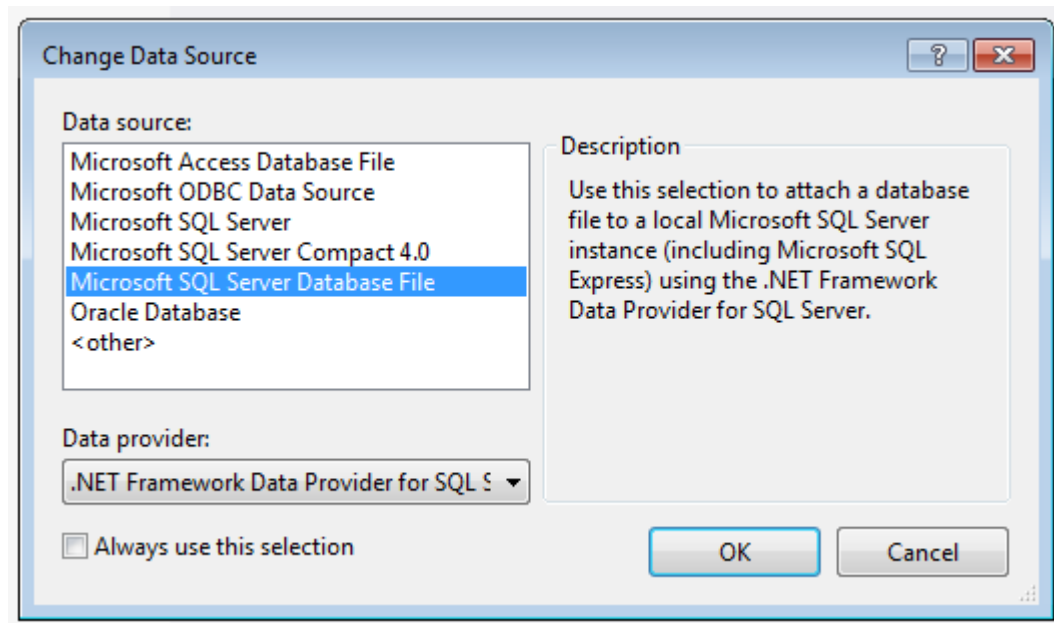
Go back to Visual Studio and find the Server Explorer to the left of the program...



Click on the button Connect to Database...

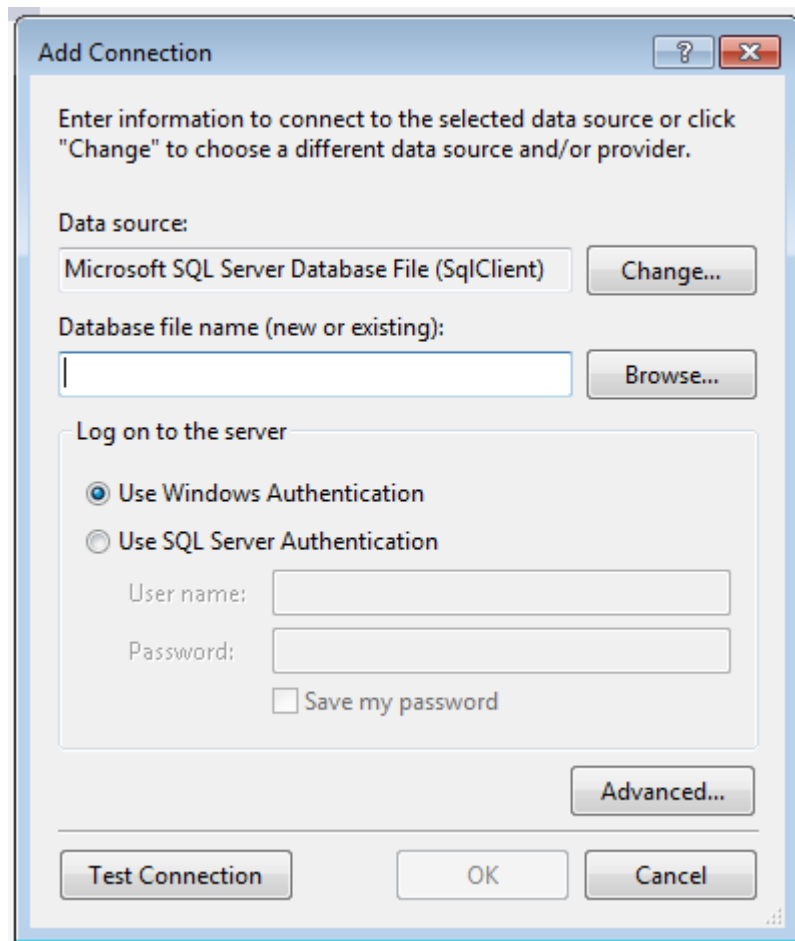


You should see the following dialogue box.

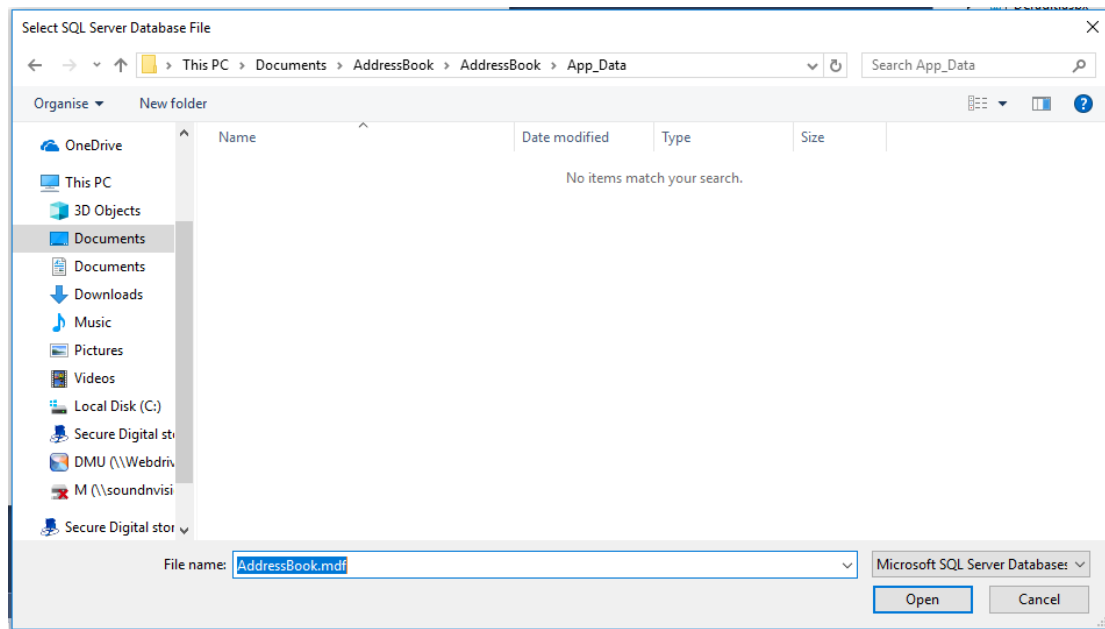


Select "Microsoft SQL Server Database File and press OK.

You should next see the following screen...



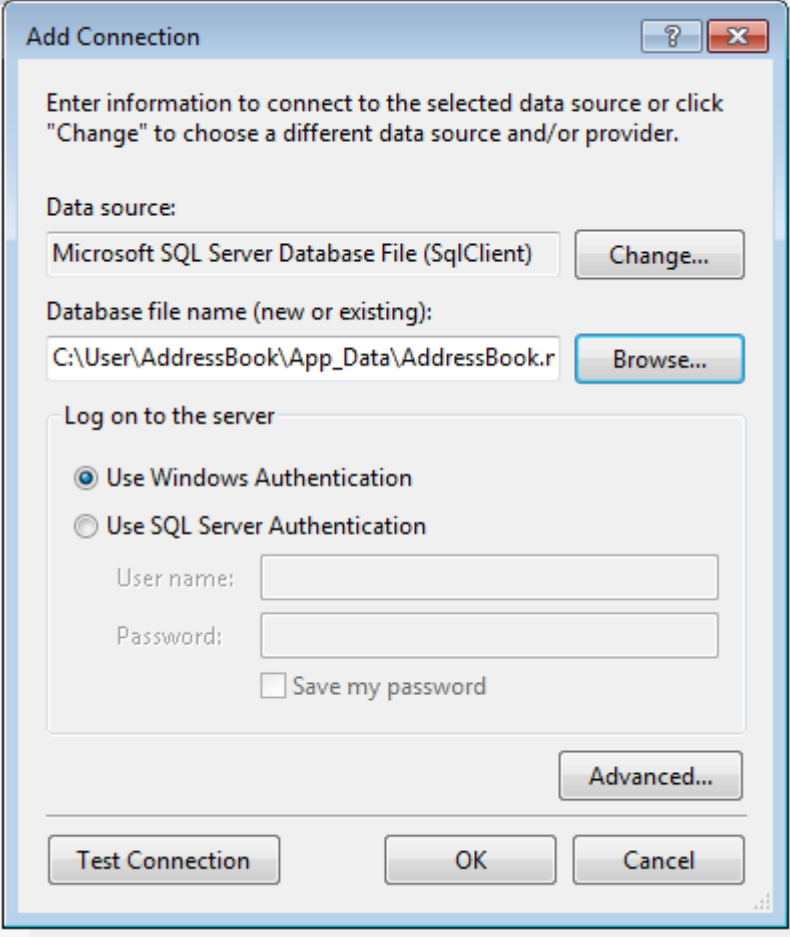
Click on Browse and navigate to the App_Data folder you created above...



Type the name of the database file you want to create, in this case AddressBook.mdf

Press Open.

On the next screen press OK



The image shows a Windows-style dialog box titled "Add Connection". It has a standard title bar with a question mark icon and a close button (X). The main text inside the dialog reads: "Enter information to connect to the selected data source or click 'Change' to choose a different data source and/or provider." Below this, there are two sections. The first section is labeled "Data source:" and contains a text box with the text "Microsoft SQL Server Database File (SqlClient)" and a "Change..." button to its right. The second section is labeled "Database file name (new or existing):" and contains a text box with the path "C:\User\AddressBook\AppData\AddressBook.r" and a "Browse..." button to its right. Below these sections is a group box titled "Log on to the server". Inside this group box, there are two radio buttons: "Use Windows Authentication" (which is selected) and "Use SQL Server Authentication". Below the radio buttons are two text boxes labeled "User name:" and "Password:". Below the "Password:" text box is a checkbox labeled "Save my password". To the right of the "Log on to the server" group box is an "Advanced..." button. At the bottom of the dialog, there are three buttons: "Test Connection", "OK", and "Cancel".

Add Connection

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:

Microsoft SQL Server Database File (SqlClient) **Change...**

Database file name (new or existing):

C:\User\AddressBook\AppData\AddressBook.r **Browse...**

Log on to the server

☒ Use Windows Authentication

☐ Use SQL Server Authentication

User name:

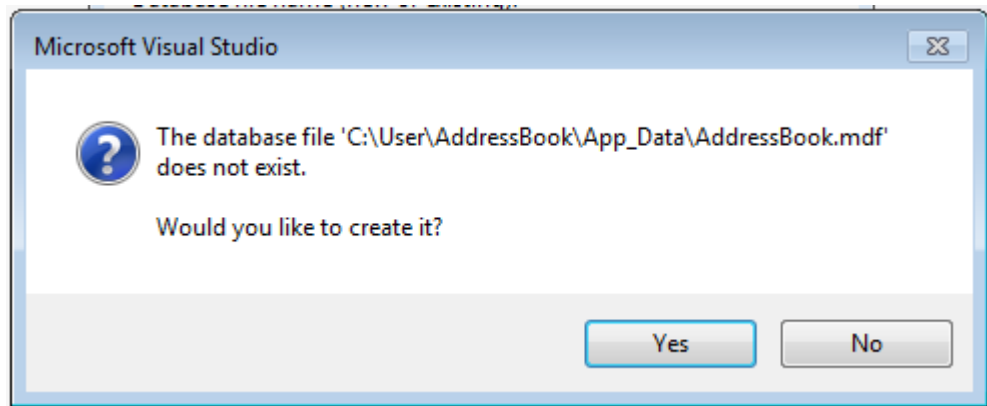
Password:

☐ Save my password

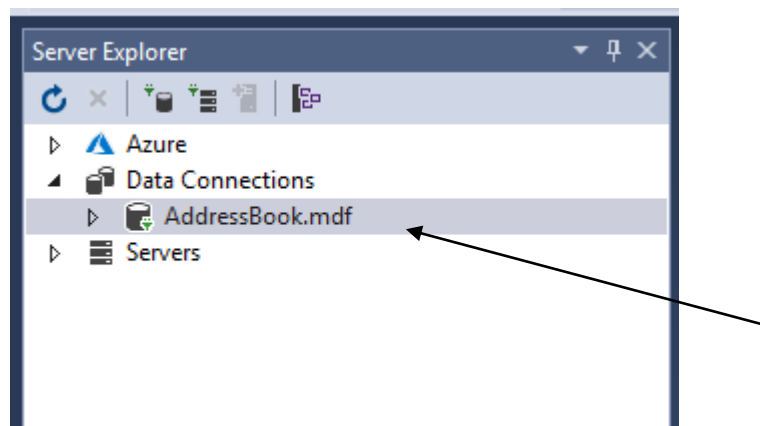
Advanced...

Test Connection **OK** **Cancel**

Visual Studio should now ask if you want to create the database file...

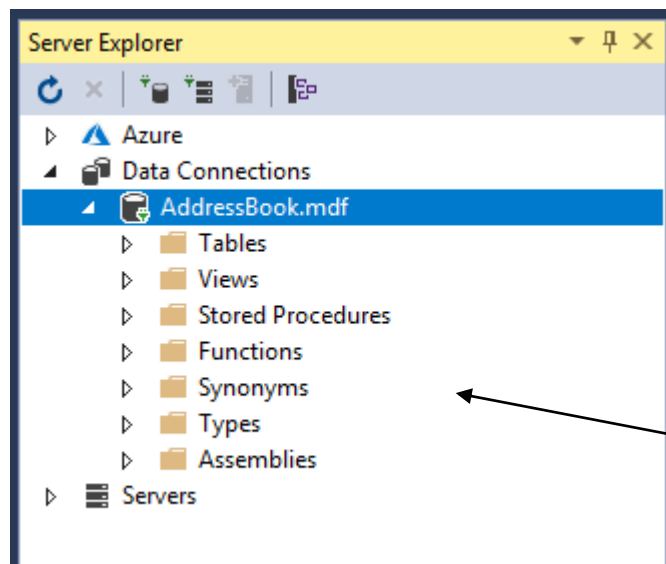


You should also be able to access it via the server explorer on the left of the program...



The server explorer is the tool in Visual Studio we use to modify the database structure.

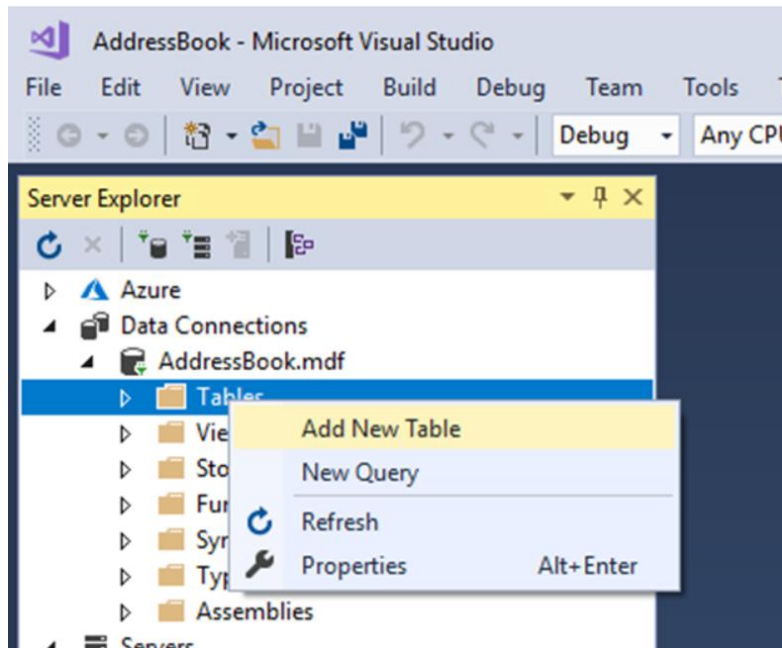
Notice the little triangle to the left of the database name. If you click on this you will be able to see the structure of the database...



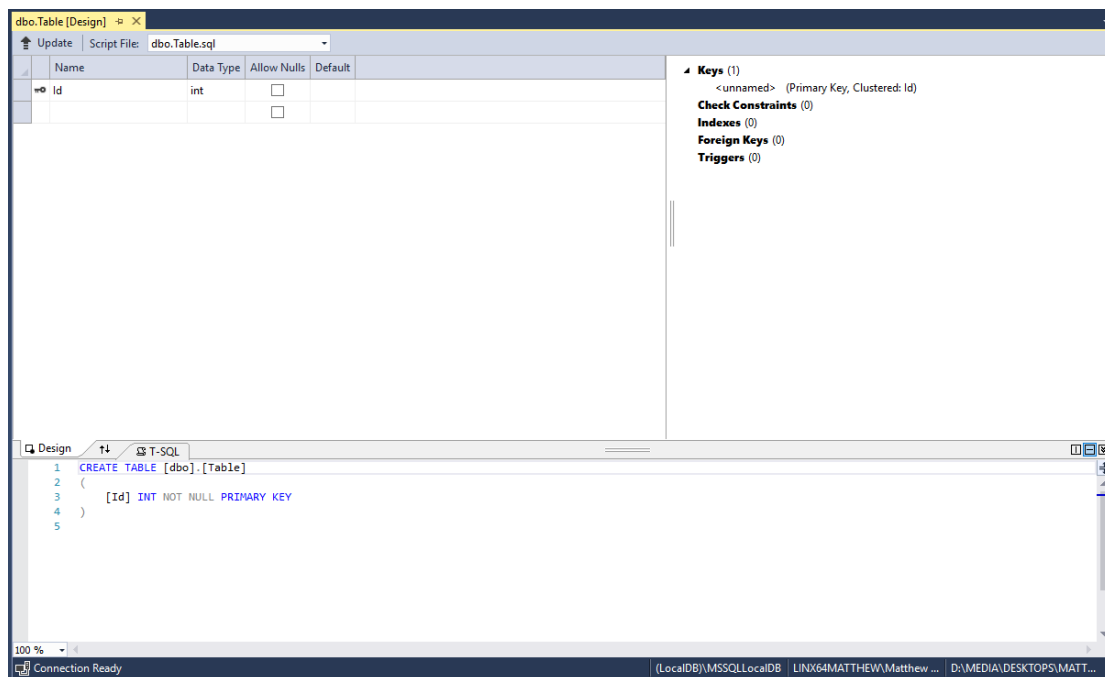
Creating the Main Table

(You really need to take a little time at this stage thinking about your system's design. You can change things later but the more planning you do now the easier it will be later. Take a little time at this stage to rough out your class diagram and entity relationship diagram!)

Right click on the entry that says Tables and select Add New Table...

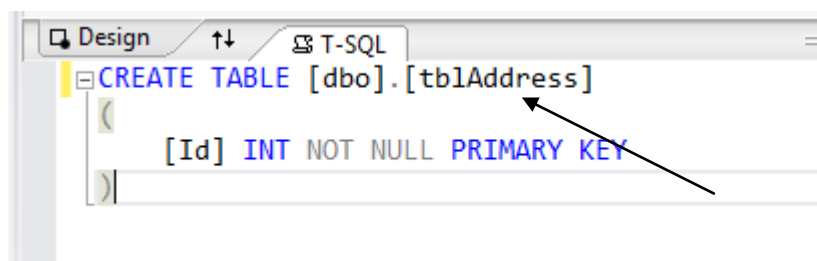


The following table design editor should appear allowing you to create the new table...



There are two ways of creating the new table. At the top of the screen is a point and click editor which we shall make most use of. At the bottom of the screen is the code view which we shall use a tiny bit.

The first thing to do is to set the name of the table. In the code at the bottom of the window change the default table name from “Table” to “tblAddress” like so...



The next step is to create the fields for the table in the top section like so...

	Name	Data Type	Allow Nulls	Default
	AddressNo	int	<input type="checkbox"/>	
	HouseNo	varchar(6)	<input checked="" type="checkbox"/>	
	Street	varchar(50)	<input checked="" type="checkbox"/>	
	Town	varchar(50)	<input checked="" type="checkbox"/>	
	PostCode	varchar(9)	<input checked="" type="checkbox"/>	
	CountyCode	int	<input checked="" type="checkbox"/>	
	DateAdded	date	<input checked="" type="checkbox"/>	
	Active	bit	<input checked="" type="checkbox"/>	
			<input type="checkbox"/>	

The column called Name identifies what the name of the data stored in the field. The column called data type identifies what sort of data we are allowed to store in that field.

For example,

Int	short for Integer meaning whole numbers. E.g. 1,2,3,-40, 100 NOT 4.5, 55.8 as these are decimal numbers.
Varchar(6)	Any combination of letters or numbers up to 6 characters
Date	Any valid data
Bit	True or False only

When creating field names, it is always a good idea to consider the following:

Avoid spaces

Avoid unusual characters

Use pascal naming convention

Avoid certain reserved words

AddressNo is better than Address No

~ ! etc..

AddressNo clearer than addressno

date, password (+ others)

Now that the fields have been set up we need to do two more things.

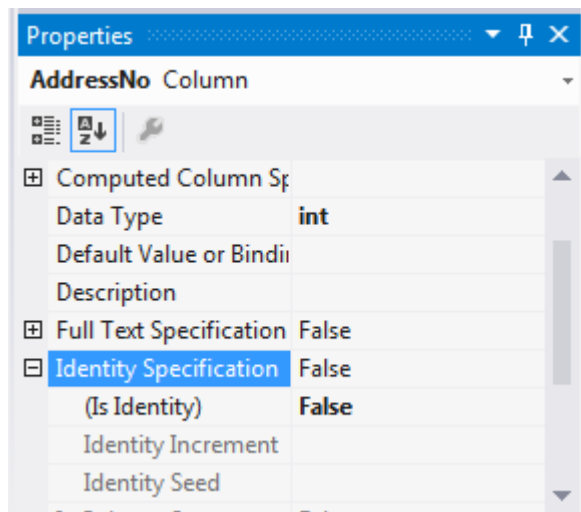
The field AddressNo is going to be the primary key for this table and will be used to uniquely identify each record in the table.

The first step is to make sure this value is unique. To do this we set it as the identity of the table so that the database will generate this value.

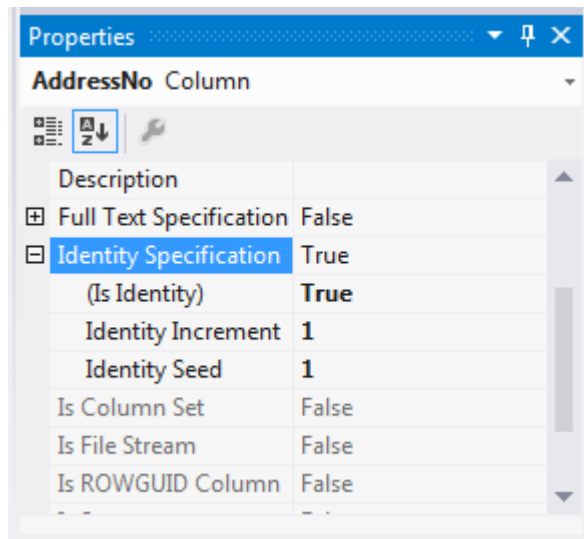
Click once in the name of the field to select it...

	Name	Data Type	Allow Nulls	Default
	AddressNo	int	<input type="checkbox"/>	
	HouseNo	varchar(6)	<input checked="" type="checkbox"/>	

In the bottom, right hand corner of the program are the properties for the field. Find the property called Identity Specification and expand it...



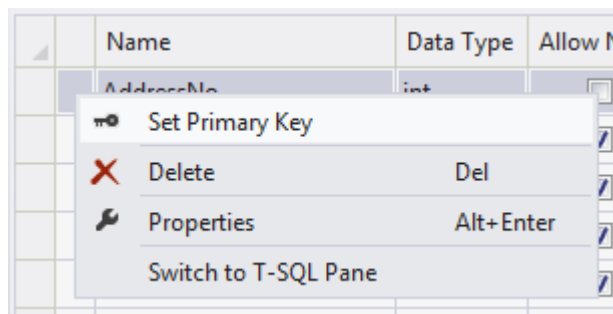
Change “Is Identity” from False to True like so...



This has set the field such it starts at 1 (identity seed) and for every new record the numeric value increases by 1 (identity increment).

The last step is to set this field as the Primary Key for this table.

Right click to the left of the field name and select Set Primary Key...

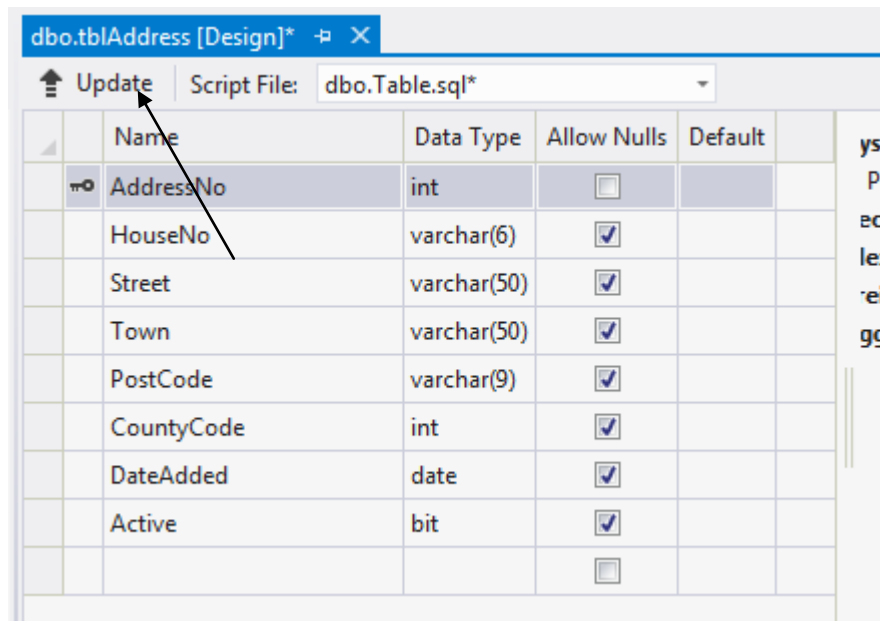


The definition for the table should now look like this...

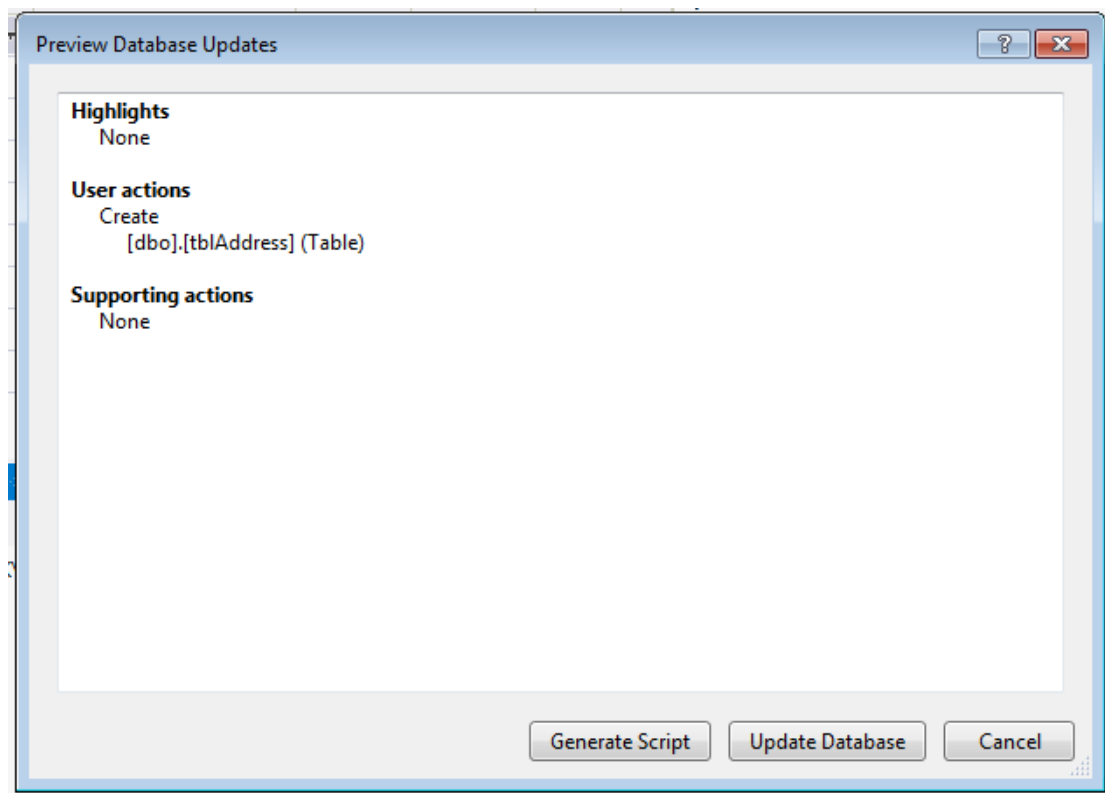
```
CREATE TABLE [dbo].[tblAddress] (  
    [AddressNo] INT NOT NULL IDENTITY,  
    [HouseNo] VARCHAR (6) NULL,  
    [Street] VARCHAR (50) NULL,  
    [Town] VARCHAR (50) NULL,  
    [PostCode] VARCHAR (9) NULL,  
    [CountyCode] INT NULL,  
    [DateAdded] DATE NULL,  
    [Active] BIT NULL,  
    CONSTRAINT [PK_tblAddress] PRIMARY KEY ([AddressNo])  
);
```

Now that the code to create the table is set up we need to create the table.

In Visual Studio press the button that says Update...

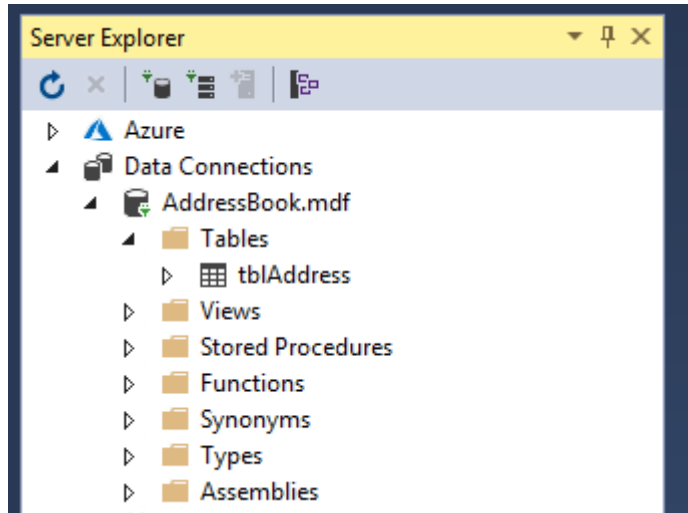


After a second or two the following screen should appear...



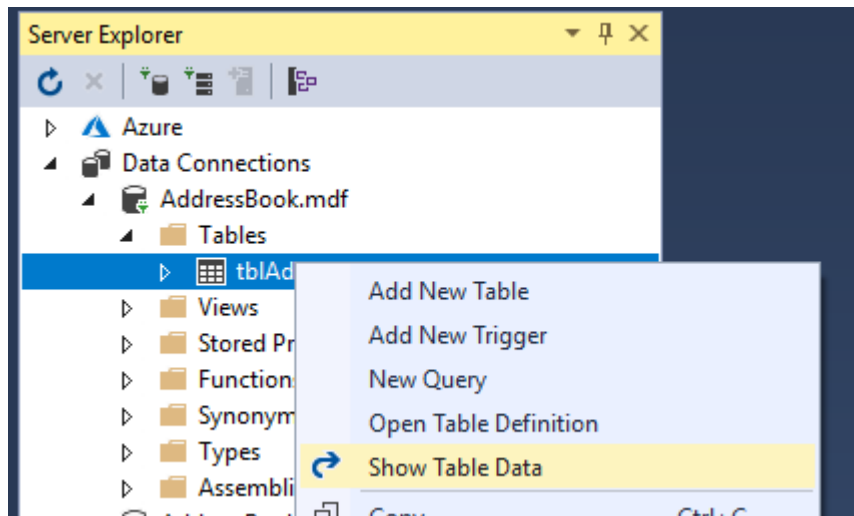
To create the new table, press Update Database.

Close the table definition and you should now be able to explore the new table in the server explorer...



The next step is to add some data to your table to make sure it is working.

In the server explorer right click on the name of the table and select show table data.



This will give you a screen allowing you to add new data to the table.

Notice as you add the data the value for the AddressNo field is updated automatically by the database...

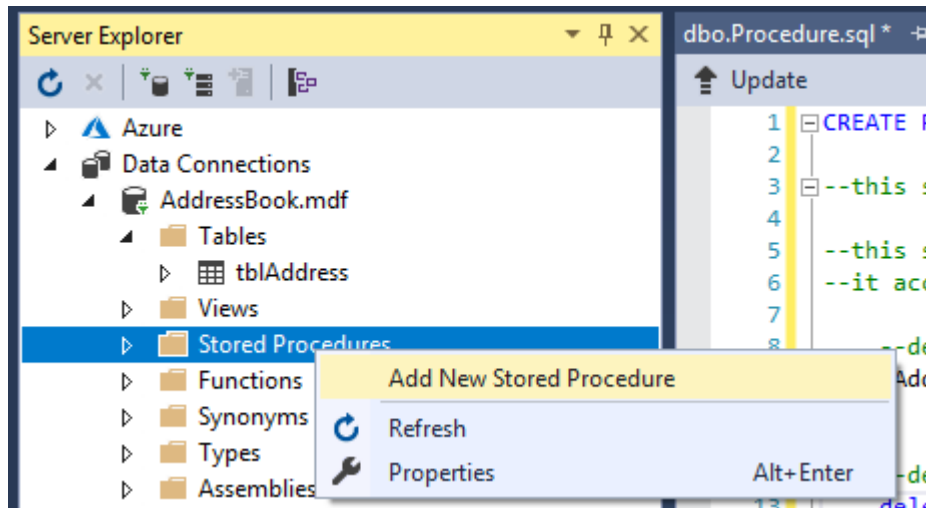
	AddressNo	HouseNo	Street	Town	PostCode	CountyCode	DateAdded	Active
▶	2	1	Some Street	Leicester	LE1 1BE	35	07/09/2012	True
	3	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
	4	33	High Street	Leicester	LE1 6FG	35	07/08/2012	True
	5	22	The Road	Nottingham	N19 6EF	48	07/08/2012	True
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Add a few records so that we have some data for the next stage of the work.

Creating the Delete Stored Procedure

Once the table has been created we need to add the delete stored procedure to make sure that we can delete records.

In the server explorer right click on the section for stored procedures and select add new stored procedure...



The first step is to give the stored procedure a name. To do this we will follow what is called a naming convention. The convention we will use here takes the following format...

sproc_table(s) to be acted upon_type of action

for example, sproc_tblAddress_Delete

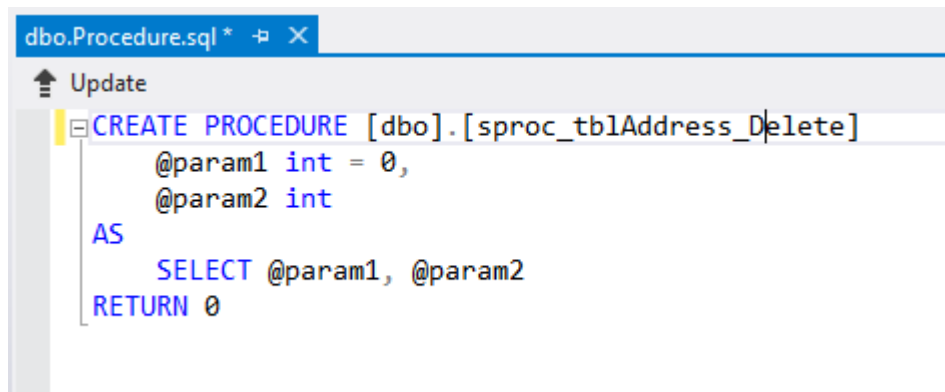
To translate...

sproc	identifies this as a stored procedure
tblAddress	is the name of the table we are using
Delete	identifies what we are doing to the table

We shall use a lot of naming conventions on this module.

Naming conventions are designed to make our lives as programmers easier. The computer doesn't care what conventions we use we can name objects whatever we like. It is important to use names for objects that make sense to us human beings as it makes our lives a lot easier.

To name the stored procedure modify the code like so...

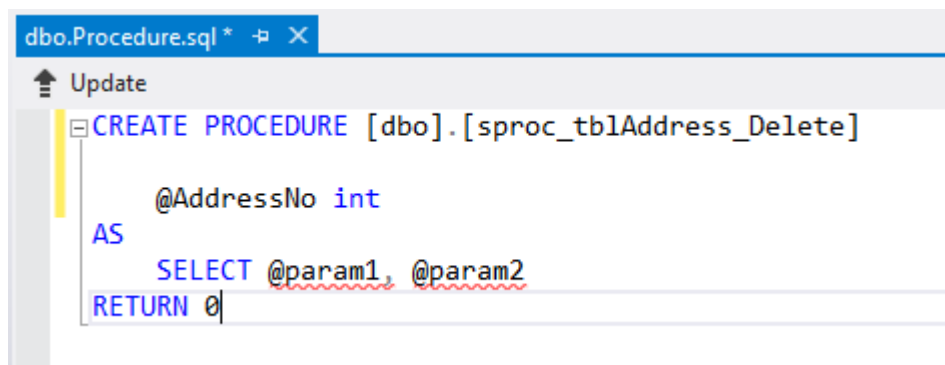


```
dbo.Procedure.sql *  X
Update
CREATE PROCEDURE [dbo].[sproc_tblAddress_Delete]
    @param1 int = 0,
    @param2 int
AS
    SELECT @param1, @param2
RETURN 0
```

The next step is to create the parameters for the stored procedure so we may send it some data. We shall add one parameter @AddressNo.

The @ symbol identifies it as a parameter, the name is AddressNo.

Modify the code like so...



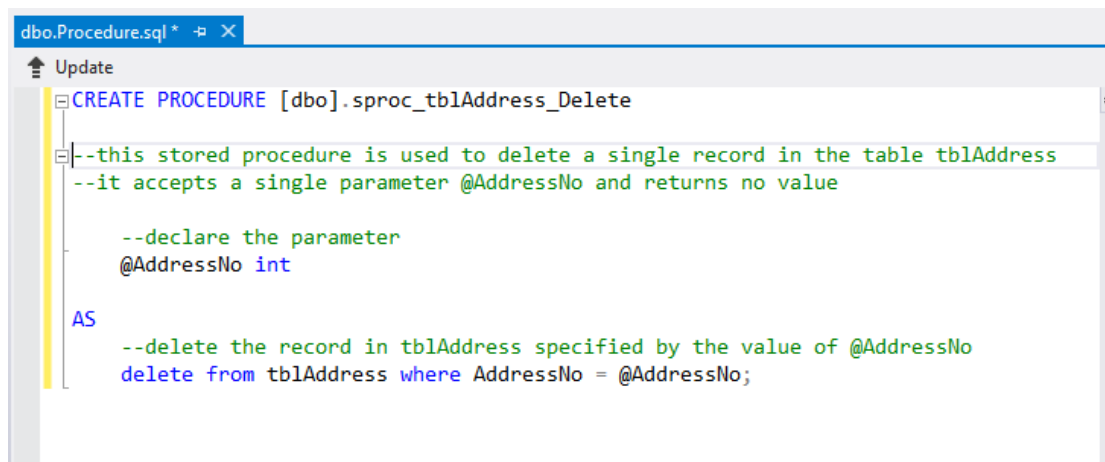
```
dbo.Procedure.sql *  X
Update
CREATE PROCEDURE [dbo].[sproc_tblAddress_Delete]
    @AddressNo int
AS
    SELECT @param1, @param2
RETURN 0
```

The last thing to do is to create some SQL to make the query do something.

Type the following SQL into the query...

```
delete from tblAddress where AddressNo = @AddressNo;
```

The last thing is to add some comments so that it's easier to understand what is going on...



The screenshot shows a SQL Server Enterprise Manager window titled 'dbo.Procedure.sql *'. The 'Update' button is visible. The SQL script being edited is as follows:

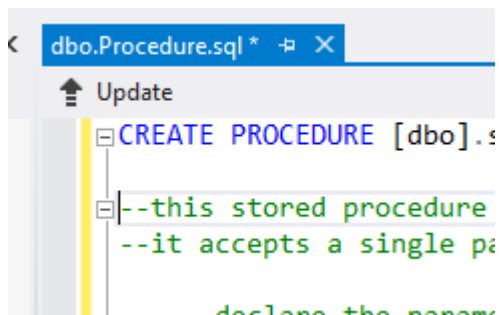
```
CREATE PROCEDURE [dbo].sproc_tblAddress_Delete
--this stored procedure is used to delete a single record in the table tblAddress
--it accepts a single parameter @AddressNo and returns no value

    --declare the parameter
    @AddressNo int

AS
    --delete the record in tblAddress specified by the value of @AddressNo
    delete from tblAddress where AddressNo = @AddressNo;
```

To create the stored procedure, follow the same procedure for creating the table.

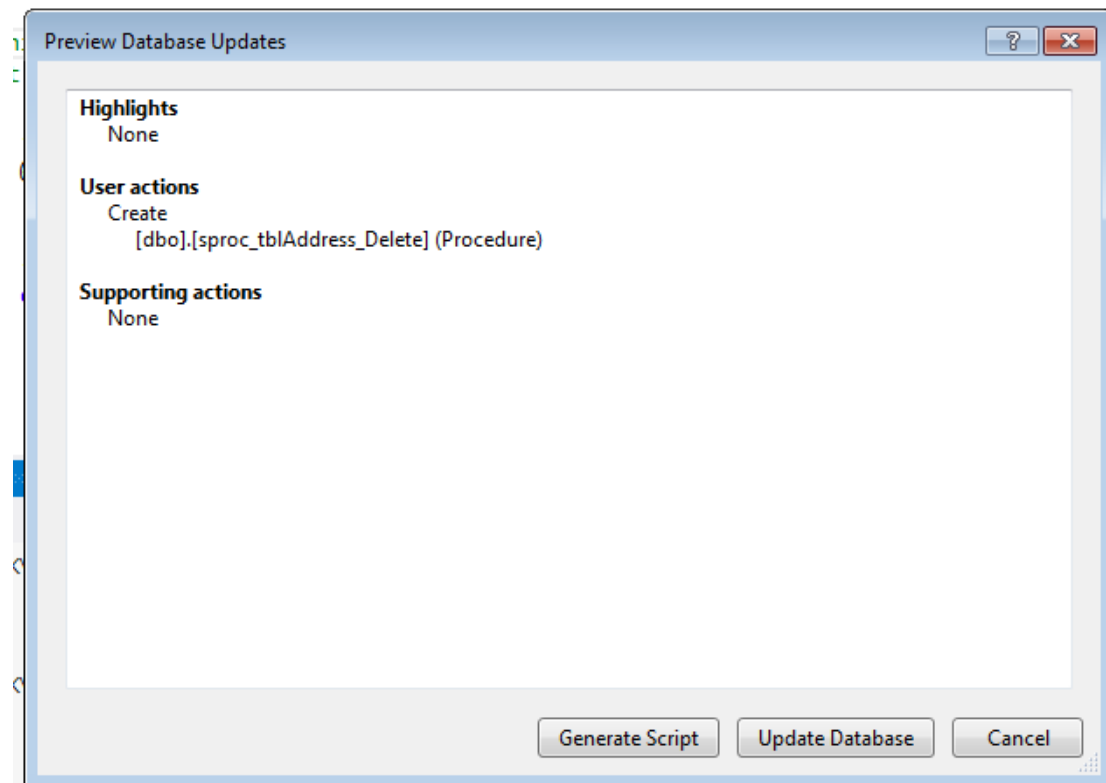
Press the update button...



This is a partial screenshot of the same SQL script editor. It shows the beginning of the script:

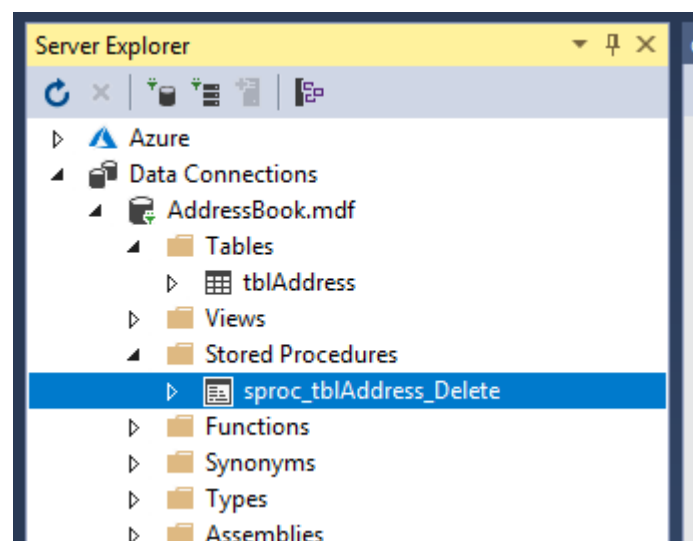
```
CREATE PROCEDURE [dbo].s
--this stored procedure
--it accepts a single pa
    declare the param
```


Wait for the script to be generated...



Select update database.

You should now be able to see the stored procedure in the server explorer.



If everything has been set up correctly you should be able to do two things.

1. Add a new address to the table
2. Delete the address using stored procedure

To test the stored procedure, make sure you have some test data in your table. Right click on the stored procedure and execute it to see if it deletes records correctly.

Once you have completed the work for this session you need to decide on the topic for your web site and repeat the above steps this time for your own database / topic.