LAB 1: Getting started with WebMatrix

Introduction

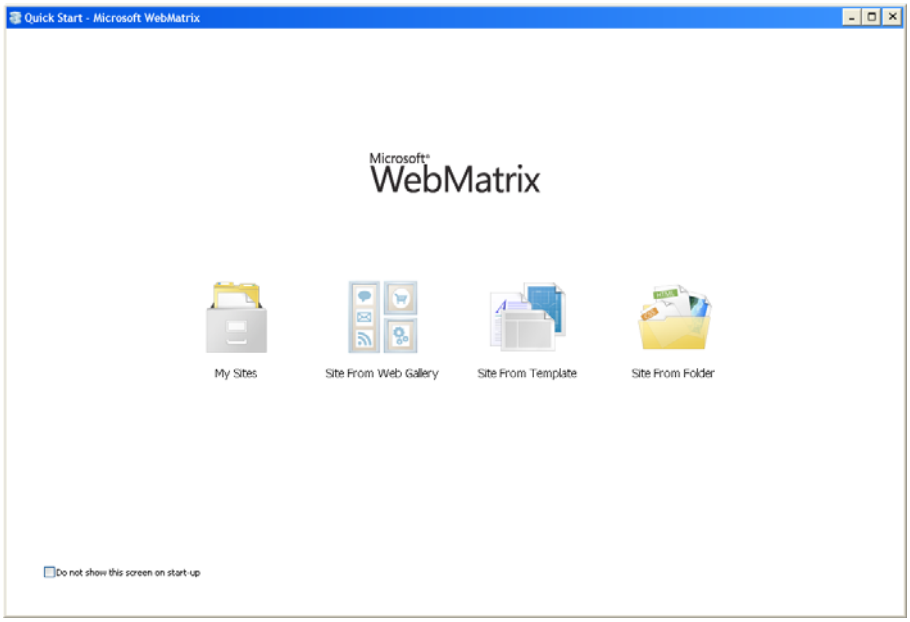
In this module you will learn the principles of database development, with the help of Microsoft WebMatrix. WebMatrix is a software application which is designed to allow you to build data-driven websites. It includes tools which can help you to create and work with a relational database.

The purpose of this first lab is simply to become familiar with WebMatrix. You can also use this lab task sheet as a reference guide when you are using WebMatrix in later lab exercises.

In the labs which follow you will put into practice the database design and development techniques which you will be learning in the lectures. This lab introduces the tools within WebMatrix which you will need to use to do this. You will create a simple database from scratch. On the way you will learn how to create database tables and queries, and simple data-driven web pages. Don't worry if you don't yet understand all the database terminology which you come across in this lab: you will learn the theory behind what you're doing as you go through the module.

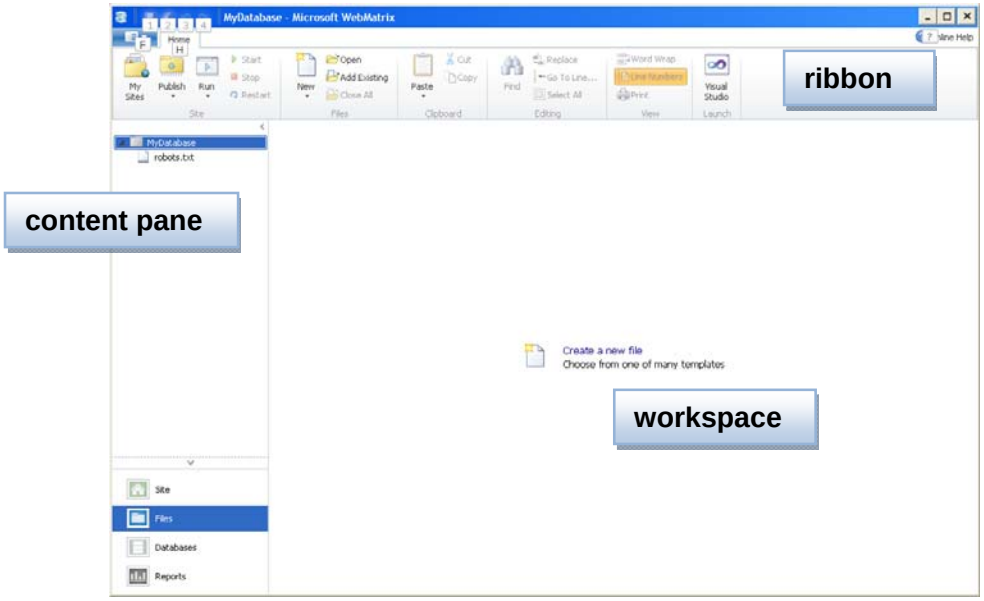
Creating a new database

1. Start WebMatrix. You should see the Quick Start screen.

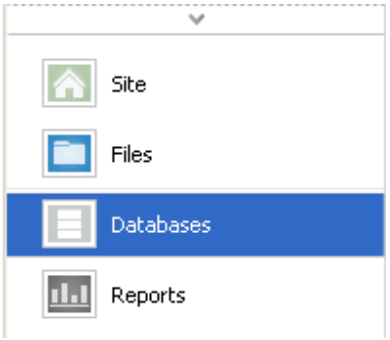


2. In WebMatrix, a database is contained within a web site, so you must first create a web site. Click **Site from Folder**. In the **Browse for Folder** box, select a location for your site. The default will probably be the My Web Sites folder, but you can change this. Click the **Make New Folder** button, and enter *My Database* as the name of the folder which will contain your site.

3. The WebMatrix window should open ready to edit the *MyDatabase* site.

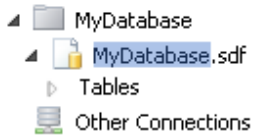


4. At the bottom left of the window, there are links which allow you to select a workspace depending on the task you are doing. You are going to create a database, so click **Databases**.



5. Click on the link which appears in the workspace to add a new database to your site.

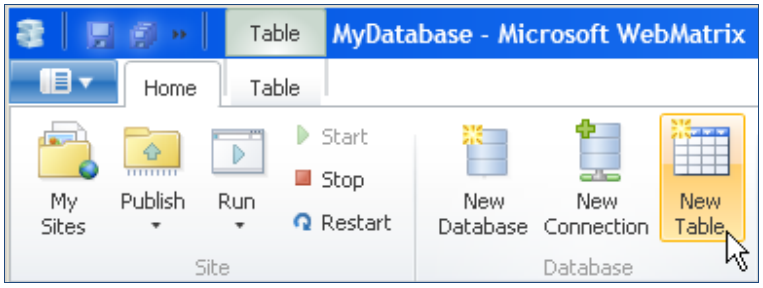
You should see a database file *MyDatabase.sdf* listed in the content pane on the left of the WebMatrix window. Files with the extension .**sdf** are Microsoft SQL Server Compact Edition database files – this is the database system we will be using in this module. You are now ready to start creating tables to store data in your database



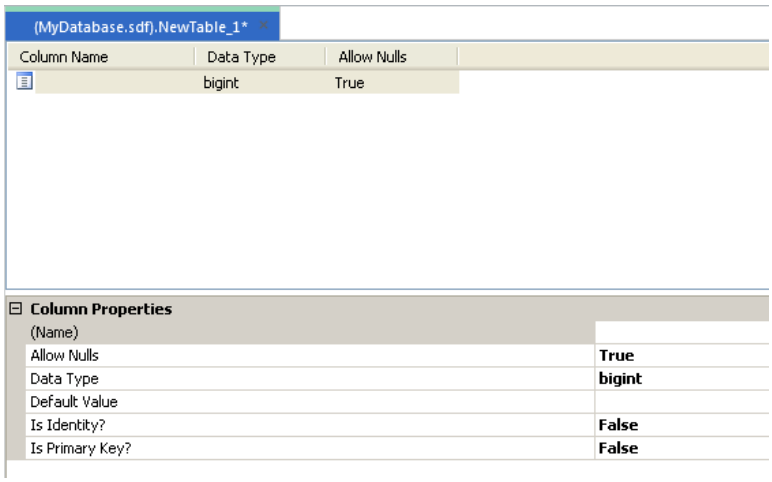
Creating a table

The data in a database is stored in tables. In this exercise you'll create two tables to store information about Formula 1 racing teams and drivers.

1. Select the **New Table** icon on the ribbon. You need to be working with the **Home** ribbon – you can select between **Home** and **Table** ribbons using the tabs just underneath the title bar.

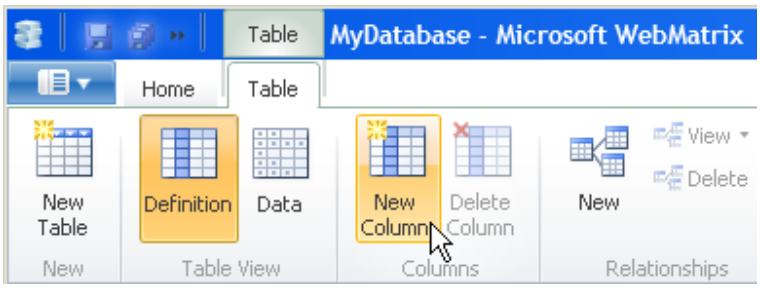


2. A new **Table window** opens in the work area, with a title New*Table_1* or something similar.



A table contains **columns**. Each column holds information of a particular type (e.g. text, number). You need to tell WebMatrix what columns you want this table to have. You enter each column name in turn, and select the data type for each column.

3. Define column names and data types as shown in the following figure. You will need to click the **New Column** icon on the ribbon each time you need to add a column. You will need to be working with the **Table** ribbon.

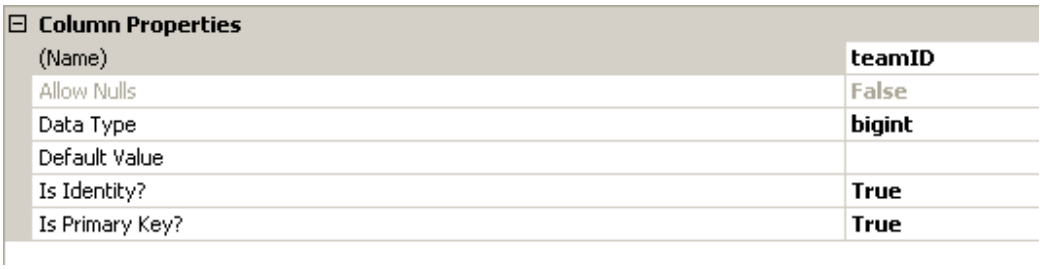


Column Name	Data Type	Allow Nulls
teamID	bigint	False
teamName	nvarchar	True
engine	nvarchar	True
nationality	nvarchar	True
points	int	True

4. The lower part of the workspace lets you set more detailed properties for each field. The *nationality* column will hold two-letter country codes, so set its **Length** to *2*. Similarly, set the **Length** of the *teamName* and *engine* columns to 15

Column Properties	
(Name)	nationality
Allow Nulls	**True**
Data Type	**nvarchar**
Default Value	**''**
Is Primary Key?	**False**
Length	**2**

5. A table should have a **primary key** column to uniquely identify each row of data. Select the *teamID* column and set **Is PrimaryKey?** to True. Also set **Is Identity?** to True – this means that values in this column will be generated automatically.

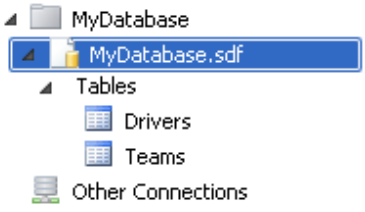


6. You should now save the table as *Teams.* Click the Save icon on the title bar and enter the table name in the dialog box.

7. Create another table called *Drivers*, with the following fields:

driverID	Identity, Primary Key
firstname	nvarchar, length 15
lastname	nvarchar, length 25
dateOfBirth	datetime
points	int
team	bigint

Both tables should now appear in the content pane

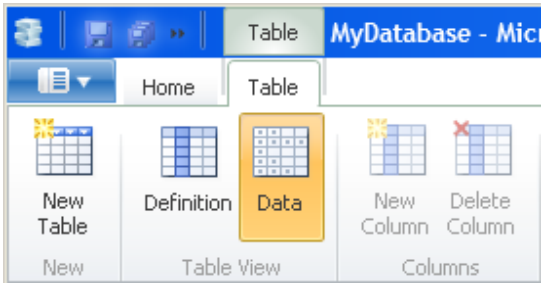


8. Close both your table windows in the workspace.

Entering data

Your tables don't have any data in them yet. You can now enter some data in the *Teams* table.

1. Double-click the *Teams* table in the content pane. The table should open in **Data** view, ready for you to enter data. If you need to switch between **Definition** (defining what columns the table has) and **Data** (entering data) views you can use the icons on the **Table** ribbon.



2. Data view shows the data in the table in **rows** and **columns**. Each row represents a team, while each column holds one piece of information for each row. There will be one blank row to start with.

3. Type the following data into the table. The *teamID* values are entered automatically when you move to the next row, so you don't need to type those. When you start typing in a new row an additional blank row is added below.

Table - (MyDatabase.sdf).Teams

	teamID	teamName	engine	nationality	points
	1	Red Bull	Renault	GB	451
	2	McLaren	Mercedes	GB	325
	3	Ferrari	Ferrari	I	254
	4	Mercedes GP	Mercedes	D	108
	5	Renault	Renault	GB	70
	6	Force India	Mercedes	IN	36
	7	Sauber	Ferrari	CH	35
	8	Toro Rosso	Ferrari	I	29
	9	Williams	Cosworth	GB	5
	10	Lotus	Renault	GB	0
	11	Hispania	Cosworth	E	0
	12	Virgin	Cosworth	GB	0
*	NULL	NULL	NULL	NULL	NULL

Modifying a table

You will now modify the *Teams* table by adding an extra column.

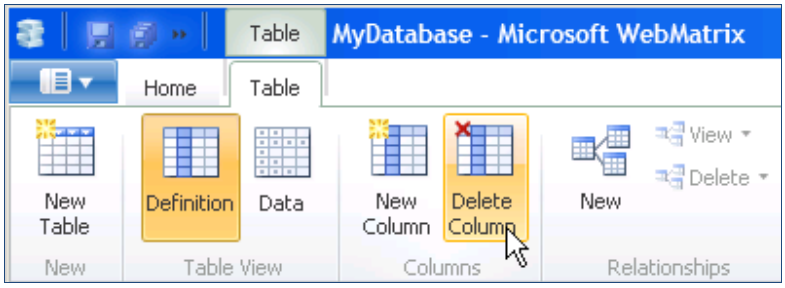
1. Open the *Teams* table if it is not already open. Switch to **Definition** view if necessary.

2. Add a new **int** column called *championships*. Save the table.

3. Switch to **Data** view and enter the following data in the new column:

championships
1
8
16
0
2
0
0
0
9
0
0
0
NULL

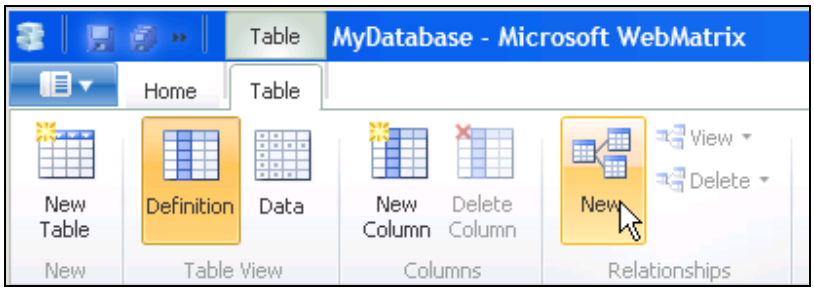
4. Let's not store that information after all. Switch back to **Definition** view, select the *championships* column and click the **Delete Column** icon on the **Table** ribbon. Close the table and choose to save the changes.



Defining a table relationship

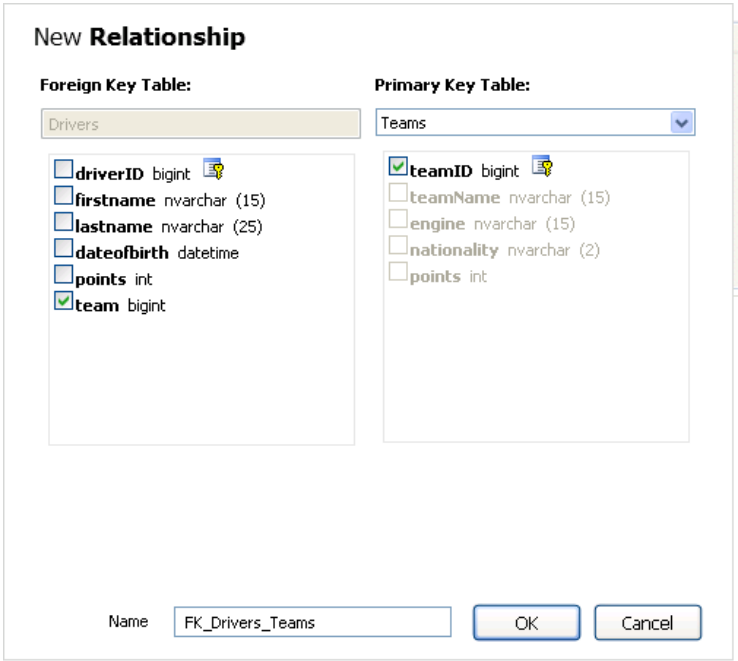
The two tables in your database are related - each driver is a member of a team. It is a good idea to define that relationship properly. If you do, the database can make sure that you don't try to make a driver belong to a team that doesn't exist. This kind of relationship is called a **Foreign Key**.

1. Open the *Drivers* table in **Definition** view and click the **New** icon on the **Relationships** section of the **Table** ribbon.

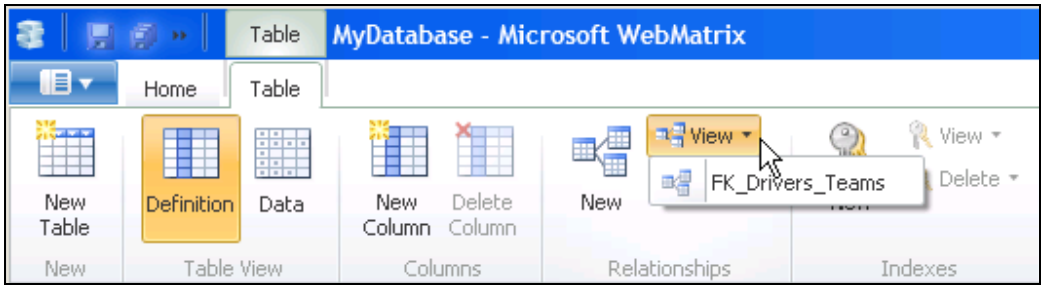


   You want to relate the *team* column in *Drivers* to the *teamID* column in *Teams*. Note that these columns both contain integer values (data type is *bigint*) – it is important that the related columns in two tables have the **same kind of data**.

2. In the **New Relationships** window, select *Teams* as the **Primary key table**, and check the *teamID* column. Also check the team column under **Foreign Key Table**, which lists the columns of *Drivers*. Click OK and then save the table.



3. Check that the foreign key has been set up correctly by clicking on the **View** icon in the **Relationships** section of the **Table** ribbon (while the *Drivers* table is open in **Definition** view). You should see one foreign key, named *FK_Drivers_Teams*, listed. Click on *FK_Drivers_Teams* to view the details of the relationship.



4. Now open the *Drivers* table in **Data** view and enter the following data. Note that the values in the *dateofbirth* column can be entered using a date format such as 3/7/87 and will be recognised by the database as dates. The *driverID* values are created automatically – do not type these.

	driverID	firstname	lastname	dateofbirth	points	team
	1	Sebastien	Vettel	03/07/1987 00:...	284	1
	2	Fernando	Alonso	29/07/1981 00:...	172	3
	3	Jenson	Button	19/01/1980 00:...	167	2
	4	Mark	Webber	27/08/1976 00:...	167	1
	5	Lewis	Hamilton	07/01/1985 00:...	158	2
	6	Felipe	Massa	25/04/1981 00:...	82	3
	7	Nico	Rosberg	27/06/1985 00:...	56	4
	8	Michael	Schumacher	03/01/1969 00:...	52	4
	9	Vitaly	Petrov	08/09/1984 00:...	34	5
	10	Nick	Heidfeld	10/05/1977 00:...	34	5

5. Try entering the following additional row. What happens? Why?

NULL	Kamui	Kobayashi	13/9/86	27	13

Modify the row so that it can be saved (*hint: Kamui Kobayashi drives for the Sauber team*).

Close all tables.

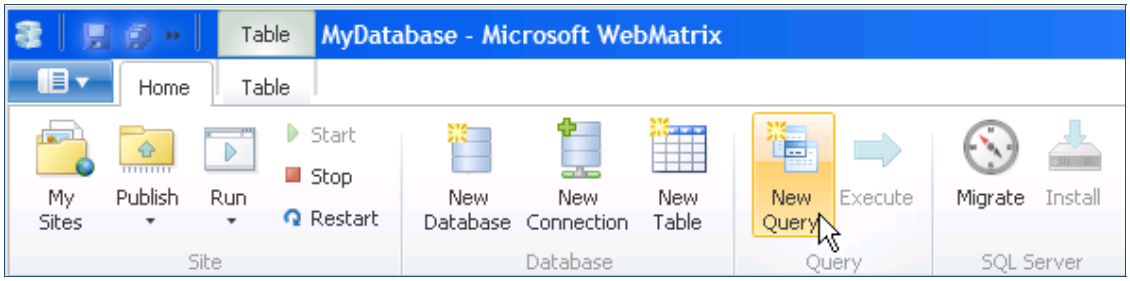
Creating a query

A query is a subset of the data in a database, and is usually used to provide answers to a specific question about the data. A query can contain data from one or more tables. You can create a query using an **SQL statement**. SQL is a (fairly) standard language which can be used to query many different types of database. You'll learn more about how to use SQL later in the module

You will create a query to answer the question:

Which drivers belong to the Ferrari team?

1. Select the **New Query** icon on the **Home** ribbon.

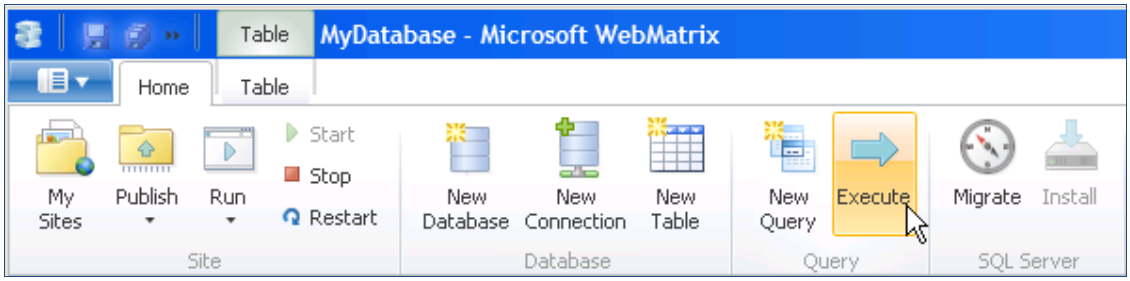


   The workspace should now have a blank window where you can type a statement.

2. Type the following text into the box. Don't worry about what this means – you will learn how to write SQL later in this module.

```
SELECT firstname, lastname
FROM Teams, Drivers
WHERE Teams.teamID = Drivers.team
AND teamname = 'Ferrari'
```

3. To see the result you need to execute the query. Click the **Execute** icon on the **Query** section of the **Home** ribbon.



The result of executing the query is shown in a tabular view below the query window.

	firstname	lastname
▶	Fernando	Alonso
	Felipe	Massa

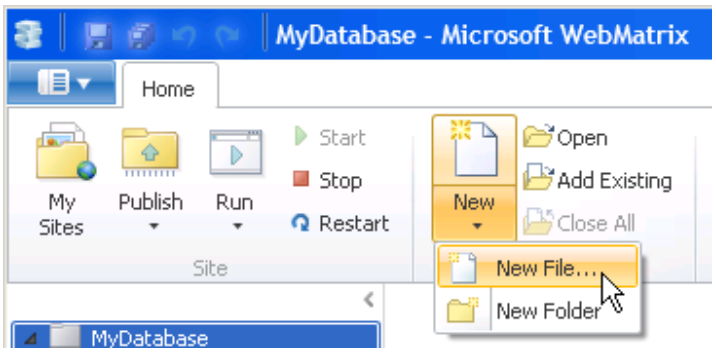
If you have made a mistake in typing the query you will probably see an error message instead of this. Read the error message carefully and try to fix the mistake, and then execute the query again.

Saving a query

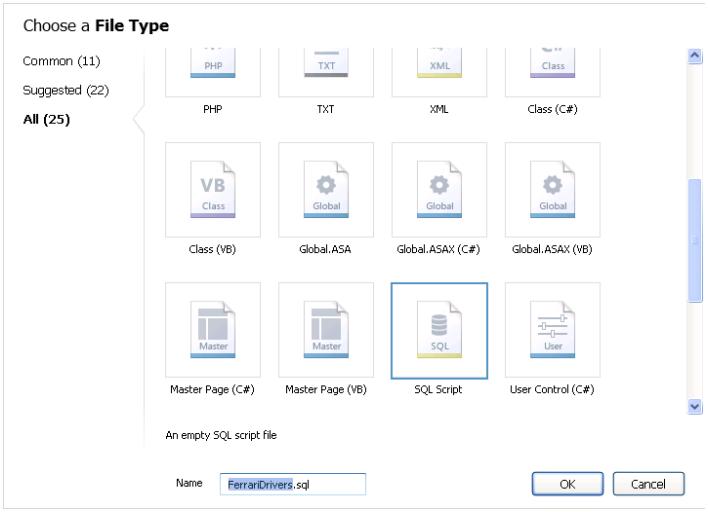
It is often useful to save an SQL statement so that you can execute it again later. WebMatrix does not make this particularly convenient, but it can be done. You can't save a query within the **Databases** workspace, but you can copy the text of your query and paste it into a file within the **Files** workspace.

1. Select all the text in the SQL query, then right-click and select **Copy** from the pop-up menu (or type CTRL-C).

2. Switch to the **Files** workspace, and click the **New** icon on the **Files** section of the **Home** ribbon and select **New File...**



3. In the **Choose a File Type** window select **SQL Script** and name the file *Ferrari Drivers.sql.* Click OK. The SQL file should appear in the content pane.



4. Paste the SQL which you copied from the query window in the **Databases** workspace into the SQL file in the **Files** workspace. Save and close the file.

5. You should make sure that you can run the query again. Open the SQL file in the **Files** workspace and select and copy the text. Switch to the **Databases** workspace, create a new query and past the text into it. Execute the query and check that you get the same result as before.

Creating an Insert Query

SQL queries can also be used to insert, update and delete data. In this exercise you'll add a new row to the *Drivers* table.

1. Create a new query as before.

2. Type the following text into the query window:

```
INSERT INTO Drivers(firstname, lastname, dateofbirth, points, team)
VALUES('Adrian', 'Sutil', '11/1/87', 24, 6);
```

   This query does not return any data. Instead, the query results area should show a reassuring message.

3. Open the *Drivers* table and check that the new row has been added. You can click the **Refresh** icon on the Table ribbon to make sure you are looking at the most up-to-date view of the data in the table.

4. Save the SQL for this query as *AddDriverQuery.sql* in the **Files** workspace.

Creating a table with SQL

You can also create a table using an **SQL query** rather than table **Definition** view. The SQL statement tells the database how to construct the table. In this exercise you'll add a new table *Cars* to the database. You'll learn how the SQL works later in this module.

1. Create a new query as before.

2. Type (carefully!) the following text into the query window:

```
CREATE TABLE Cars(
carID bigint IDENTITY PRIMARY KEY,
cartype nvarchar(10) NOT NULL,
team bigint NOT NULL,
CONSTRAINT FK_Cars_Teams FOREIGN KEY (team)
REFERENCES Teams(teamID)
);
```

3. Execute the query. A confirmation message should be displayed in the query results area.

4. Right-click **Tables** in the content pane, and select **Refresh** from the pop-up menu. The new table should now be shown in the content pane.

5. Open the *Cars* table in **Definition** view. Look at the columns and their data types. Can you see where these have been defined in the SQL? The SQL statement also sets up a foreign key relationship. Examine the foreign key relationship for the *Cars* table.

6. Save the SQL for this query as *CreateCarsTableQuery.sql* in the **Files** workspace.

7. Open the *Cars* table in **Data** view and enter the following row of data:

   carType: ***MP4/26****, team:* ***2*** (the *carID* value will be assigned automatically)

8. Switch to **Definition** view and view the relationship which has been created. Note that this relationship was defined in the SQL statement which you used to create the table.

Modifying a table with SQL – recreating the table

In this exercise you will add an extra field to the *Cars* table.

1. Close the *Cars* table if it is open. Right-click on *Cars* in the content pane and select **Delete Table**. Click **Yes** to confirm. Refresh and check that the *Cars* table no longer appears in the content pane.

2. Copy the text from *CreateCarsTableQuery.sql* into a new query. Edit the SQL by adding an extra line, as shown below:

```
CREATE TABLE Cars(
carID bigint IDENTITY PRIMARY KEY,
cartype nvarchar(10) NOT NULL,
builddate datetime NOT NULL,
team bigint NOT NULL,
CONSTRAINT FK_Cars_Teams FOREIGN KEY (team)
REFERENCES Teams(teamID)
);
```

3. Execute the query. A new *Cars* table should be shown in the content pane (remember to refresh).

4. Open the *Cars* table in **Data** view and enter the following data:

	carID	cartype	builddate	team
	1	RB7	01/04/2011 00:...	1
	2	RB7	14/04/2011 00:...	1
	3	MP4/26	01/02/2011 00:...	2
	4	MP4/26	14/03/2011 00:...	2
	5	MP4/26	03/08/2011 00:...	2

Modifying a table with SQL – altering the table

In the previous exercise you had to delete the table and all its data before you could create a modified version. If you have a lot of data in the table you don't want to do this. In this exercise you will add another extra field to *Cars* without deleting any data.

1. Close the *Cars* table if it is open. Create a new query and type the following text into the query window:

```
ALTER TABLE Cars
ADD status nvarchar(15);
```

2. Execute the query. A confirmation message should be displayed in the query results area.

3. Open the *Cars* table in Data View. Your data should still be there. There should be a new field *status* with containing no data in any of the rows. Enter data in this field as shown.

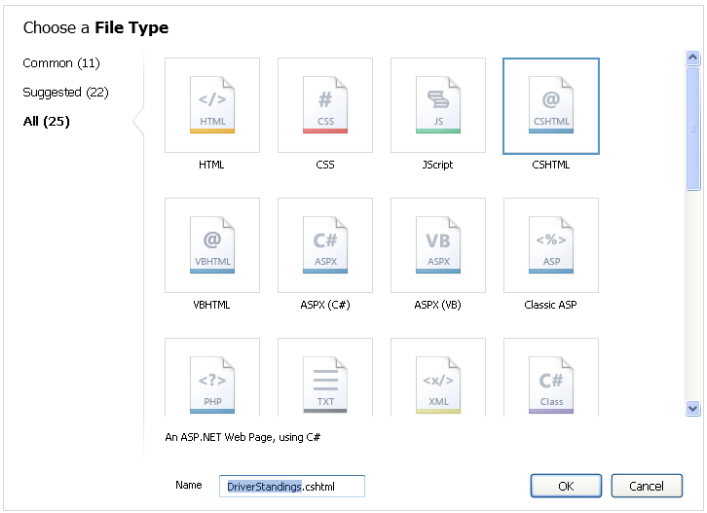
	carID	cartype	builddate	team	status
	1	RB7	01/04/2011 00:...	1	race car
	2	RB7	14/04/2011 00:...	1	test car
	3	MP4/26	01/02/2011 00:...	2	written off
	4	MP4/26	14/03/2011 00:...	2	race car
	5	MP4/26	03/08/2011 00:...	2	race car

4. Save the SQL for this query as *AlterCarsTableQuery.sql* in the **Files** workspace.

Creating a web page which displays data

WebMatrix is a tool for developers – end users of a database are more likely to interact with the data in a different way. A common situation in which users work with data is when they use a **data-driven web site**. This allows users to view and change the information in the database using a set of user-friendly web pages. In this exercise you will create a web page which displays data from the *Drivers* table.

1. Switch to the **Files** workspace. Create a **New File**, and select **CSHTML**. This file type is a web page which can include code written in the C# programming language to display data. Name the file *DriverStandings.cshtml* and click **OK**.



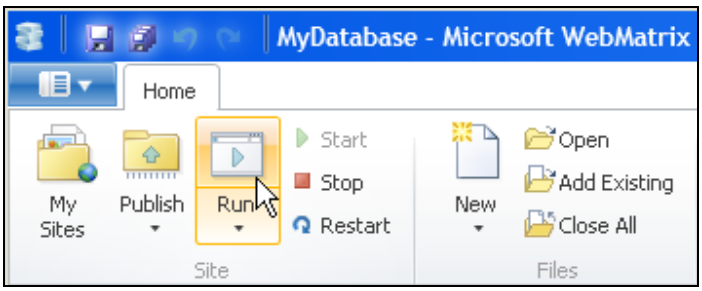
The new file should appear in the content pane and open for editing in the workspace. The file contains some HTML code which defines the layout of a web page. You can now edit the HTML and add some C# code to get the data. Don't worry about the meaning of the code just now – you will learn about these languages in this and other modules on your course.

2. Replace the code in *DriverStandings.cshtml* with the following and save the file.

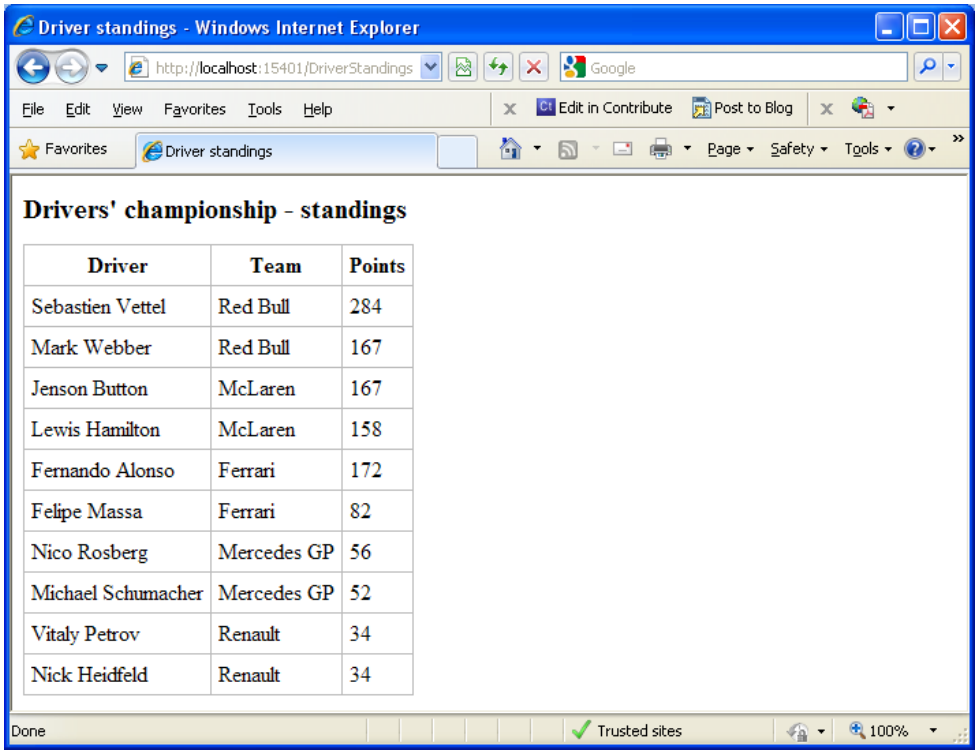
```
@{
    var db = Database.Open("MyDatabase");
    var selectQueryString =
        @"SELECT TOP 10
            Drivers.firstname,
            Drivers.lastname,
            Teams.teamname,
            Drivers.points
        FROM Drivers, Teams
        WHERE Drivers.team = Teams.teamiD";
 }

<!DOCTYPE html>
<html lang="en">
    <title>Driver standings</title>
    <style>
        h1 {font-size: 20px;}
        table, th, td {
          border: solid 1px #bbbbbb;
          border-collapse:collapse;
          padding:5px;
        }
     </style>
  </head>
  <body>
    <h1>Drivers' championship - standings</h1>
    <table>
        <thead>
            <tr>
                <th>Driver</th>
                <th>Team</th>
                <th>Points</th>
            </tr>
        </thead>
        <tbody>
            @foreach (var row in db.Query(selectQueryString)){
             <tr>
                  <td>@row.firstname  @row.lastname</td>
                  <td>@row.teamname</td>
                  <td>@row.points</td>
             </tr>
            }
        </tbody>
    </table>
  </body>
</html>
```

Now you can test the page. Note that you can't simply open the page as a file in a browser. You need to get WebMatrix to 'run' the page. If you don't do this, the data will not be displayed.

3. Click the **Run** icon on the Home ribbon.



The default browser on your computer should open, and the web page should be displayed. Be patient – this may take a few moments to display. If the web page is not displayed correctly, and you see an error page instead, check that you have entered the code correctly, and then refresh your browser to test again.



Drivers' championship - standings

Driver	Team	Points
Sebastien Vettel	Red Bull	284
Mark Webber	Red Bull	167
Jenson Button	McLaren	167
Lewis Hamilton	McLaren	158
Fernando Alonso	Ferrari	172
Felipe Massa	Ferrari	82
Nico Rosberg	Mercedes GP	56
Michael Schumacher	Mercedes GP	52
Vitaly Petrov	Renault	34
Nick Heidfeld	Renault	34

Creating a data entry form web page

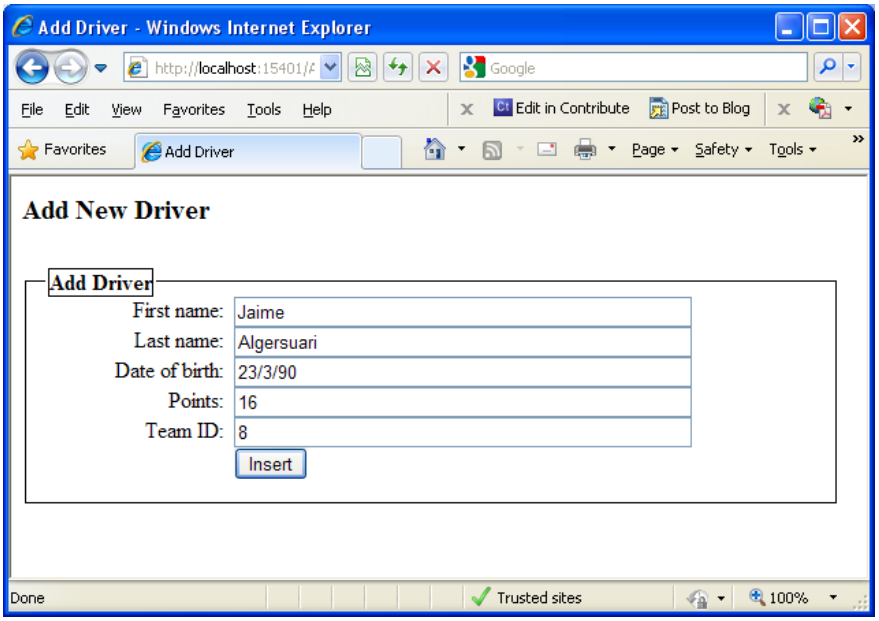
In this exercise you will create a web page which contains a form which allows the user to add a new row to the *Drivers* table.

1. In the **Files** workspace, create a new CSHTML file and save it as *AddDriver.cshtml*. Download the file *AddDriver.txt* from your module website and open it in Notepad. Copy and paste its contents into *AddDriver.cshtml*, replacing the existing HTML code. Save the file

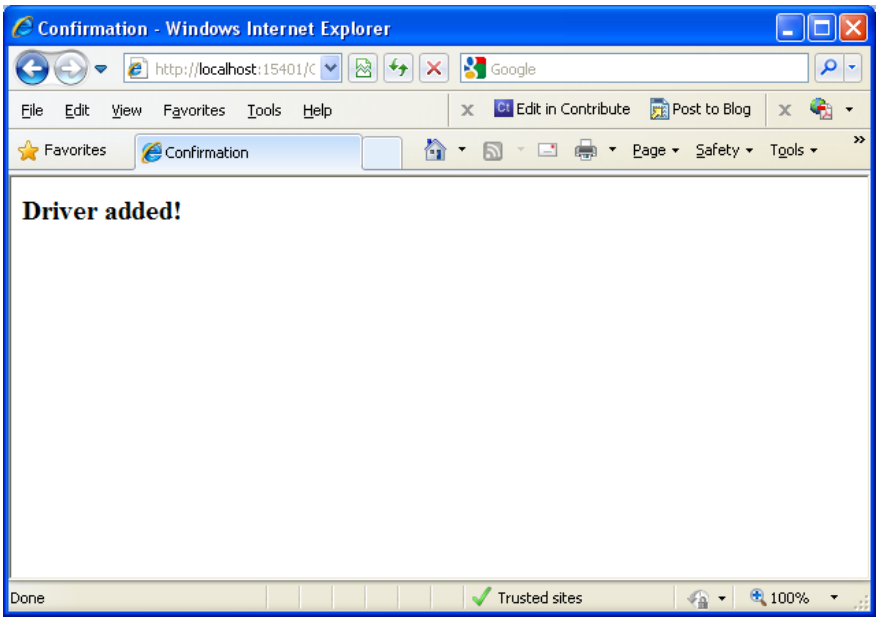
2. Create a new CSHTML file called *Confirm.cshtml.* Replace its contents with the following, then save and close the file.

```
<!DOCTYPE html>

<html lang="en">
    <head>
        <meta charset="utf-8" />
        <title>Confirmation</title>
        <style>
        h1 {font-size: 20px;}
     </style>
    </head>
    <body>
        <h1>Driver added!</h1>
    </body>
</html>
```

3. Make sure that *AddDriver.cshtml* is open and run the page. You should see a data entry form in your browser. Enter the data shown in the figure below and click **Insert**.



Your browser should be redirected to the confirmation page. If you see an error page instead, check that you have entered the code correctly, and then go back to the form page, refresh your browser and test again.



4. Switch to the **Databases** workspace in WebMatrix. Open the *Drivers* table and check that there is a new row which contains the data you entered in the web form.