

SQL Fundamentals

LEARN TO DESIGN & BUILD A WINDOWS SQL
DATABASE

STEPHEN O'CONNOR

Database

What is a database?

A file structured for the repository of data. Organized for easy retrieval, sorting, grouping, relating to other data, used to analysis information in numerous ways.

Example: Customer database

To store Customer Data, use a simple text file, not easily manageable. Why not use an Excel spreadsheet file?
No easy way to relate sales data to customer information

Relational Database Management System RDBMS

Relational Database Theory – Organizing data into tables that can be related together, this reducing redundancy and increasing the integrity of the data.

Normalization – the process of determining what information belongs in which tables to minimize redundancy and increase integrity.

Database Objects

- Tables contain
 - Columns
 - Rows or records the value of a single column and a single row is a called a “field”

Data Types

Text varchar(100)

```
Create TABLE person (  
    Name varchar(100)  
);
```

Double pie

Exact number

Int or integer unsigned integer is a positive number.

```
CREATE TABLE person (  
    AGE INTEGER UNSIGNED // expected to be a positive value  
);
```

Decimal

Decimal(precision, scope)

Decimal (10, 0)

```
CREATE TABLE Student (  
    avg_score decimal(6, 2)
```

```
);
```

The is the default (10, 0)

1234567890

(5, 2)

123.45

(8, 7)

9.999999

```
CREATE TABLE student (  
    id integer auto_increment // given to primary keys  
)
```

date 'YYYY-MM-DD' // format of string ``

time '[H]HH:MM:SS

datetime 'YYYY-MM-DD HH:MM:SS'

```
CREATE TABLE order (  
    order_date date  
);
```

Booleans 0 is false non-0 or 1 is true

Boo

Boolean tinyint(1)

```
CREATE TABLE order (  
    fulfilled Boolean // true or false 0 or 1  
);
```

Data Integrity

Keeping data valid, of the correct data type, etc., so that it is usable for its intended purpose. Customers can be moved separate tables

What Needs to be Defined

- The data type for each column in a table
- The maximum size of data that will be stores in the column
- The nullability of a column

In the customer table the credit limit field is a real data type, this is so that the user has to enter a numerical value, so that calculations can be made using the numerical data. The real data type is approximate numerical. The customer since is a date time data type, only dates can be enter into this field. Null fields allow records to be saved to the database, to maintain data integrity.

Primary Key

A field or combination of fields that make a given row unique in the database. A way of differentiating each row in a table when all other fields might be duplicated in other rows in the same table. It can never be null, and must be set when the record is created and never changed.

CONSTRAINT

A rule that is enforced on the table.

CONSTRAINT check_id CHECK (customerID BETWEEN 0 and 10000)

Identity column an attribute that will be automatically increment a field of data in each successive row added to the database. Typically used on the primary key fields to make them unique.

Foreign key

Relate one or more rows in one table to a record in another table that shares the same value in its primary key. To check who made the order use customerID as a foreign key to relate the tables in the database. Data is linked, order is linked to the customer who created the order, i.e. query the database; how much a customer has spent. Adding a FK constraint prevents deletions in the customer table to create orphaned rows in the order table. FK constraints enforce "Referential Integrity".

Customers			
KEY	customerID	int (11)	
	firstName	varchar(50)	
	lastName	varchar(50)	
	address	varchar(200)	Null
	city	varchar(200)	Null
	county	char(10)	Null
	zip	char(10)	Null
	creditLimit	real	
	customerSince	datetime	

Orders	
KEY	orderID orderDate orderAmount paymentType
FK	customerID

CRUD – CREATE, Read /SELECT, UPDATE, DELETE

CREATE - Tables are created using the CREATE TABLE statement

CREATE TABLE Customer (inside specify the table details, database schema);

SELECT - Read rows a table

SELECT firstName, lastName FROM Customer;

Or read all rows from a table

SELECT * FROM Customer;

DELETE – Delete rows from table

DELETE FROM Customer – deletes all rows from Customer table

UPDATE - Used to change data in existing rows in the table.

Specific the table after the keyword UPDATE Customer SET firstName = 'Stephen'

Selecting records

WHERE

Reading data is the main thing you do with a database. WHERE can be used to filter records.

Only select a row where the given value is true.

SELECT * FROM Customer WHERE firstName = 'Stephen' // selects a single row from the database.

Also can select the column value

SELECT * FROM Customers WHERE CustomerID = 3;

SELECT * FROM Customers WHERE CustomerID != 3;

SELECT * FROM Customers WHERE lastName = CustomerID > 5;

More than one condition AND /OR keywords can chain where clauses

SELECT * FROM Customers WHERE CustomerID = 1 OR CustomerID = 2;

IN To specify multiple possible values for a column or to shorten the above statement

SELECT * FROM Customers WHERE CustomerID IN (1, 2); // a list of value in parenthesis

LIKE Search for a pattern % 0 or more of any other character

SELECT * FROM Customer WHERE firstName LIKE 'S%'

SELECT * FROM Customers WHERE lastName = 'OConnor' AND CustomerID > 5;

SELECT * FROM Customer WHERE lastName = 'OConnor' OR CustomerID > 5;

Comparsion		Logical	
=	Equals	ALL	NOT
>	Greater Than	AND	OR
<	Less Than	ANY	SOME
>=	Greater than or equal	BETWEEN	IN
<=	Less than or equal	EXISTS	LIKE
<>	Not Equal to	BETWEEN	Between an inclusive range
!=	Not Equal to		

Order

Customers table sort data by date use ORDER BY. Can be used with ASC or DESC.

customerID	firstName	lastName	address	city	county	zip	creditLimit	customerSince
1	Ste	OConnor	Lucan	Dublin	Dublin	12	100	15/08/2002
2	John	Henry	Limetree	Limerick	Limerick	10	500	11/03/2004
3	Henry	Chinaski	Blossom	Tralee	Kerry	38	1000	02/11/2003
4	Harvey	Pecar	Cherrywood	Galway	Galway	3	75	24/07/2015
5	Brendan	Behan	Ringsend	Dublin	Dublin	11	5000	03/12/2009

SELECT * ORDER BY customerSince

customerID	firstName	lastName	address	city	county	zip	creditLimit	customerSince
1	Ste	OConnor	Lucan	Dublin	Dublin	12	100	15/08/2002
3	Henry	Chinaski	Blossom	Tralee	Kerry	38	1000	02/11/2003
2	John	Henry	Limetree	Limerick	Limerick	10	500	11/03/2004
5	Brendan	Behan	Ringsend	Dublin	Dublin	11	5000	03/12/2009
4	Harvey	Pecar	Cherrywood	Galway	Galway	3	75	24/07/2015

Group

```
SELECT customerID, zip FROM Customers GROUP BY customerID
```

customerID	zip
1	12
2	10
3	38
4	3
5	11

Join

```
SELECT * FROM Orders INNER JOIN Customers;
```

```
SELECT firstName, lastName, orderAmount FROM Customers INNER JOIN Orders ON CustomerID = CustomerID;
```

View

A view stores the select statement as a table.

```
SELECT
```

```
SELECT COUNT(*) FROM Customers;
```

Download and PowerShell, run SQLite3.exe create new database.

```
sqlite>../sqlite3.exe PatsClothesShop.db
```

```
Customers      Order
```

```
sqlite> CREATE TABLE Customers(  
    costumerID INT PRIMARY KEY    NOT NULL,
```

```

firstName      CHAR(50) NOT NULL,
lastName       CHAR(50) NOT NULL,
address        VARCHAR(200),
city           CHAR(10),
county         CHAR(10),
creditLimit..... REAL,
costomerSince  DATETIME
);

```

```

sqlite> CREATE TABLE Order(
    orderID INT PRIMARY KEY    NOT NULL,
    orderDate    DATETIME NOT NULL,
    orderAmount  REAL,
    paymentType  INT,
    customerID   INT          NOT NULL
);

```

```

sqlite>.tables
Customers      Order

```

```

sqlite>.header on
sqlite>.mode column
sqlite>.timer on

```

Insert 5 customers like below

```

INSERT INTO Customers (firstName, lastName, address, city, county, creditLimit,
costomerSince)
VALUES (1, 'Paul', 'Murphy' 32, 'Apt 1', 'Dublin', 15000.00, '2007-01-01 10:00:00' );

```

Databinding

Utilizing Databinding in a C# Win forms App. Data sets working with the System.Data Namespace (aka ADO.NET) Working with the Visual Studio's IDE's tools, windows, etc. Microsoft Visual Studio 2013 makes it easily to create databases for beginners and experts.

Databinding the user interface controls, retrieve and display data from a data source without requiring the programmer to worry about all the programmatic details of this process. Each user interface control has different properties that can be bound to a data source.

ADO = ActiveX Data Objects

User interface controls must be data binding "aware", ADO.NET(System.Data) classes support data binding.

- ADO.NET creates a connection to a data source (database)

- ADO.NET manages the conversation (requests and responses) between your application and the database.
- ADO.NET manages the data that is retrieved from the response to the database query.
- BindingSource manages the connection between the user interface controls and the underlying data set retrieved from the database. Provides an application interface to reduce learning curve for the end user. Restrict access to the database to maintain security. To control the presentation of the data. Maintain the integrity of the data. Practice

ADO.NET does a lot of the grunt work, it is not necessary to know all about ADO.NET.

Write application interface

- Reduce the learning curve for the end user
- Restrict access to the database to maintain security
- To control the presentation of the data – website, content management system
- To maintain the integrity of the data

SQL Server

A high end relational database management system.

SQL server 2013 Express Edition, similar power, but intended for smaller projects.

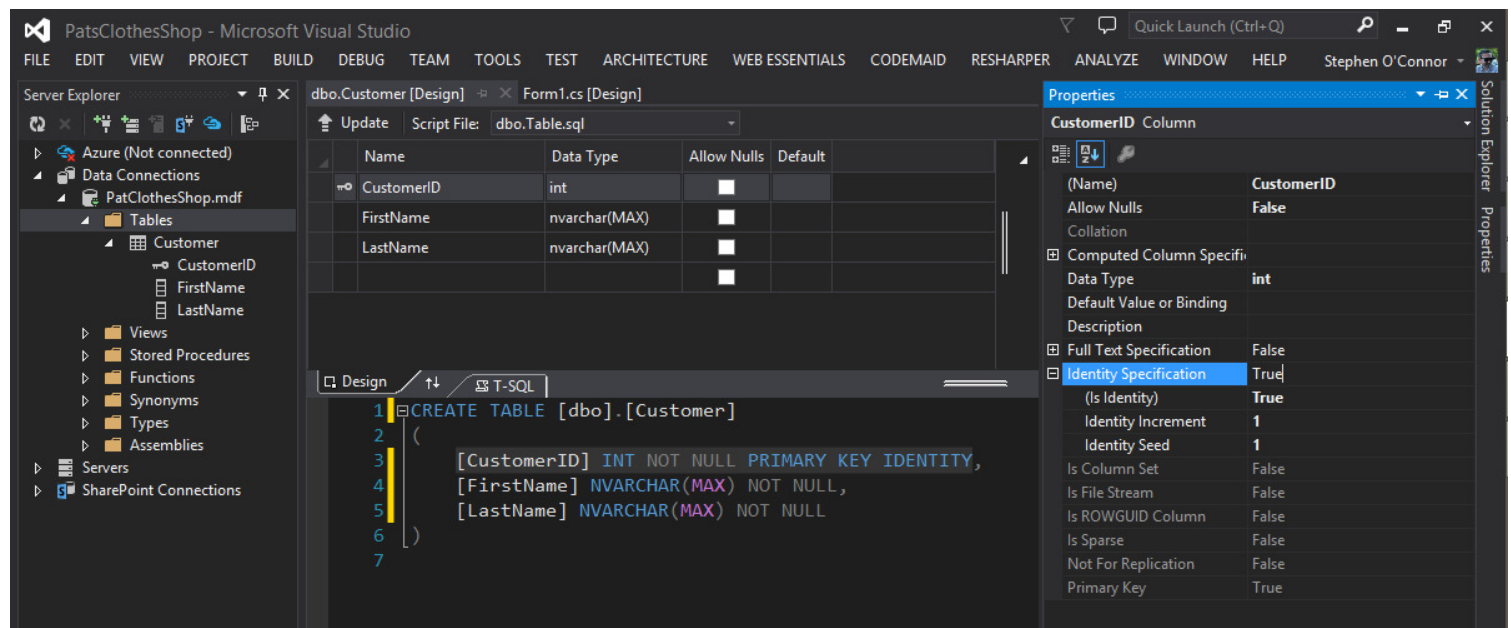
In Visual Studio 2013 download the latest SQL Server Data Tools, if already not installed.

Learn by doing

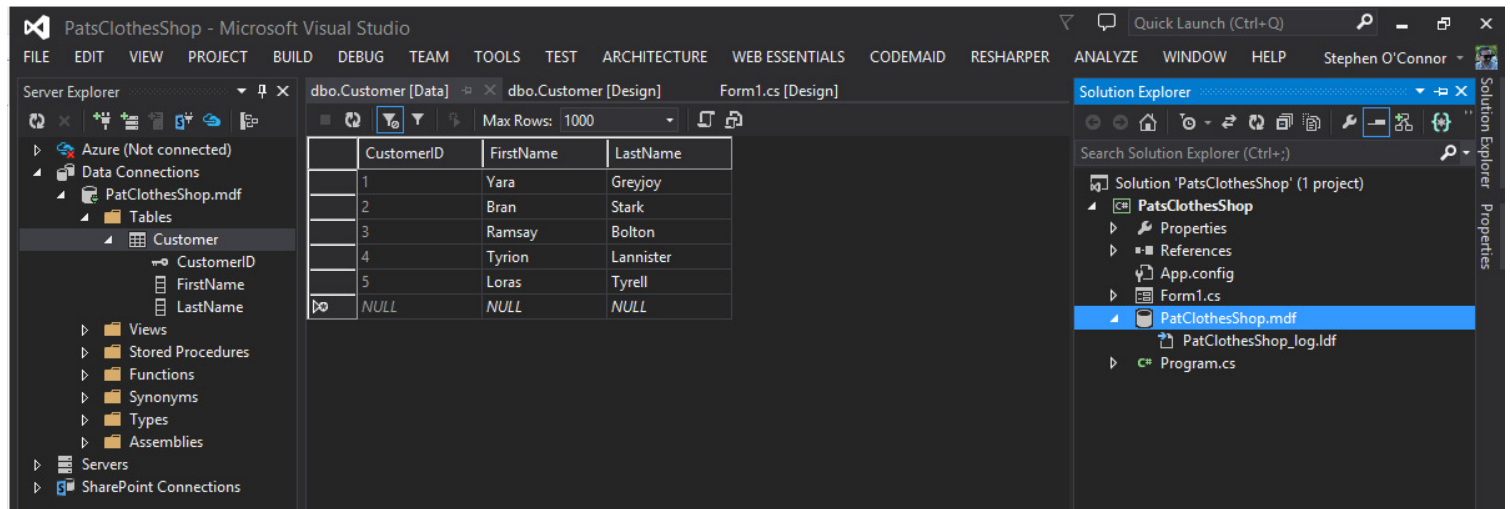
Create a new project and a database called PatClothesShop

Create a project called Pats

Add a table called customer with the data below.

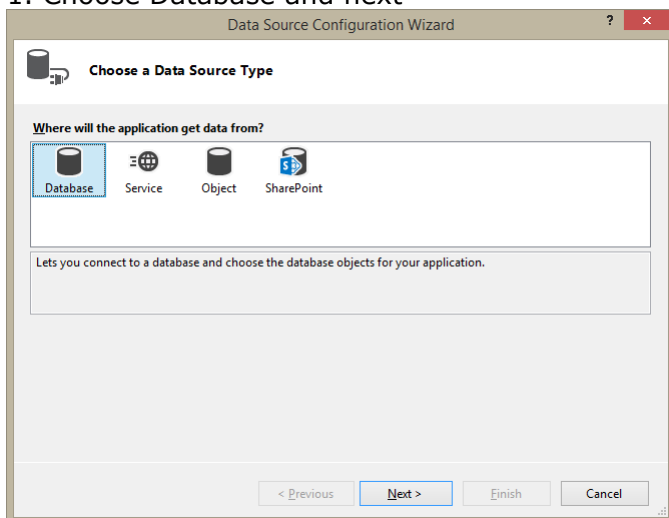


Go to show table data in the Server Explorer add five persons.

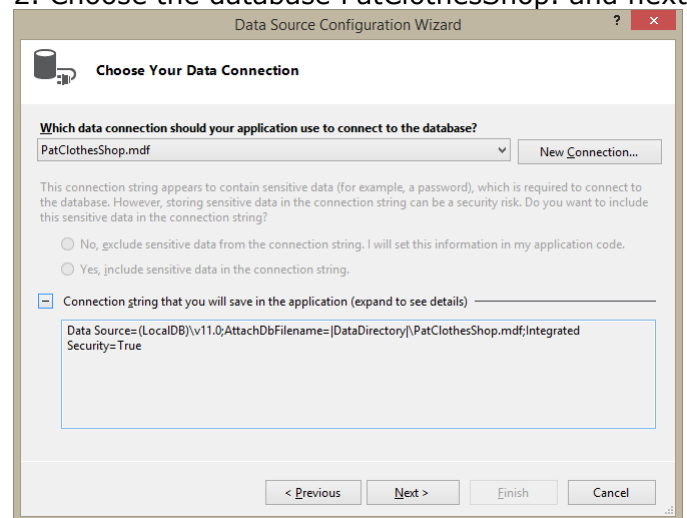


On the menu bar, choose View, Other Windows, Data Sources (or choose the Shift+Alt+D keys). Follow the steps.

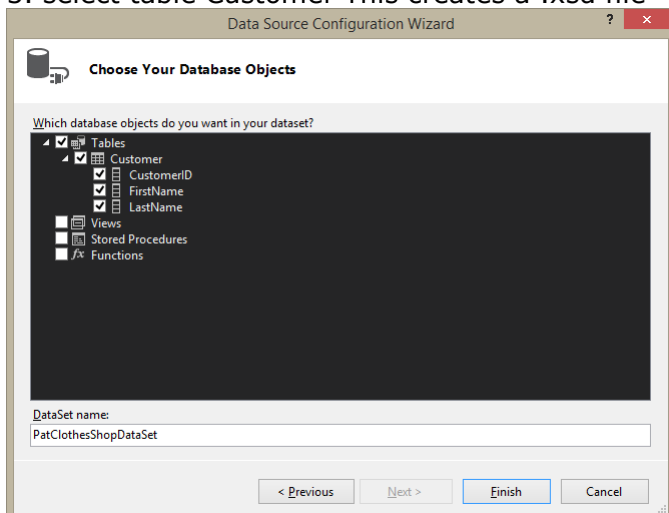
1. Choose Database and next



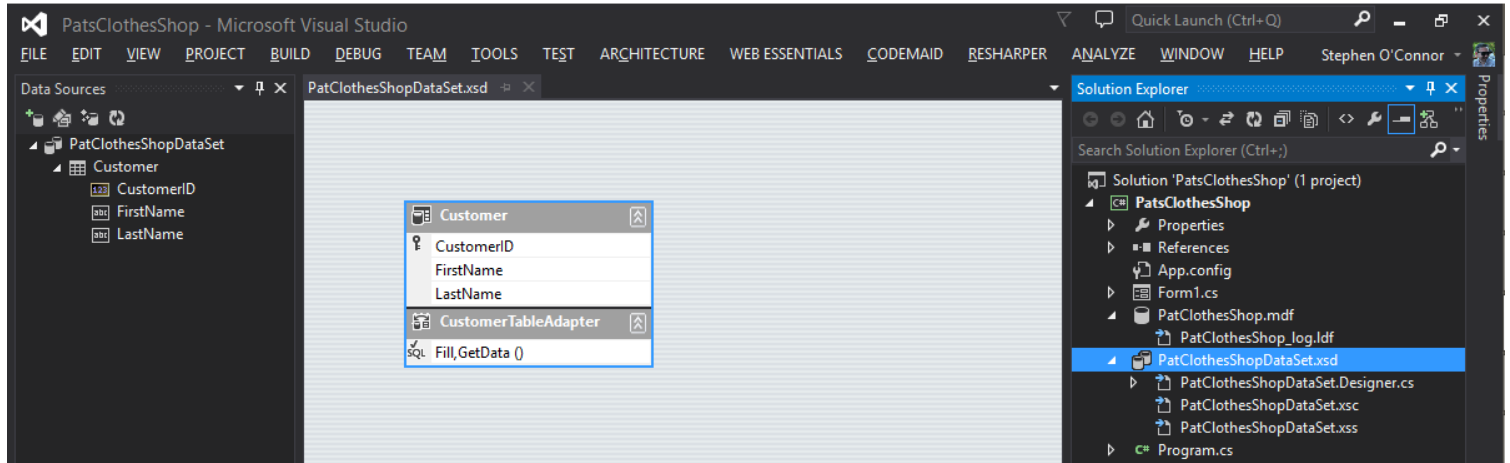
2. Choose the database PatClothesShop. and next



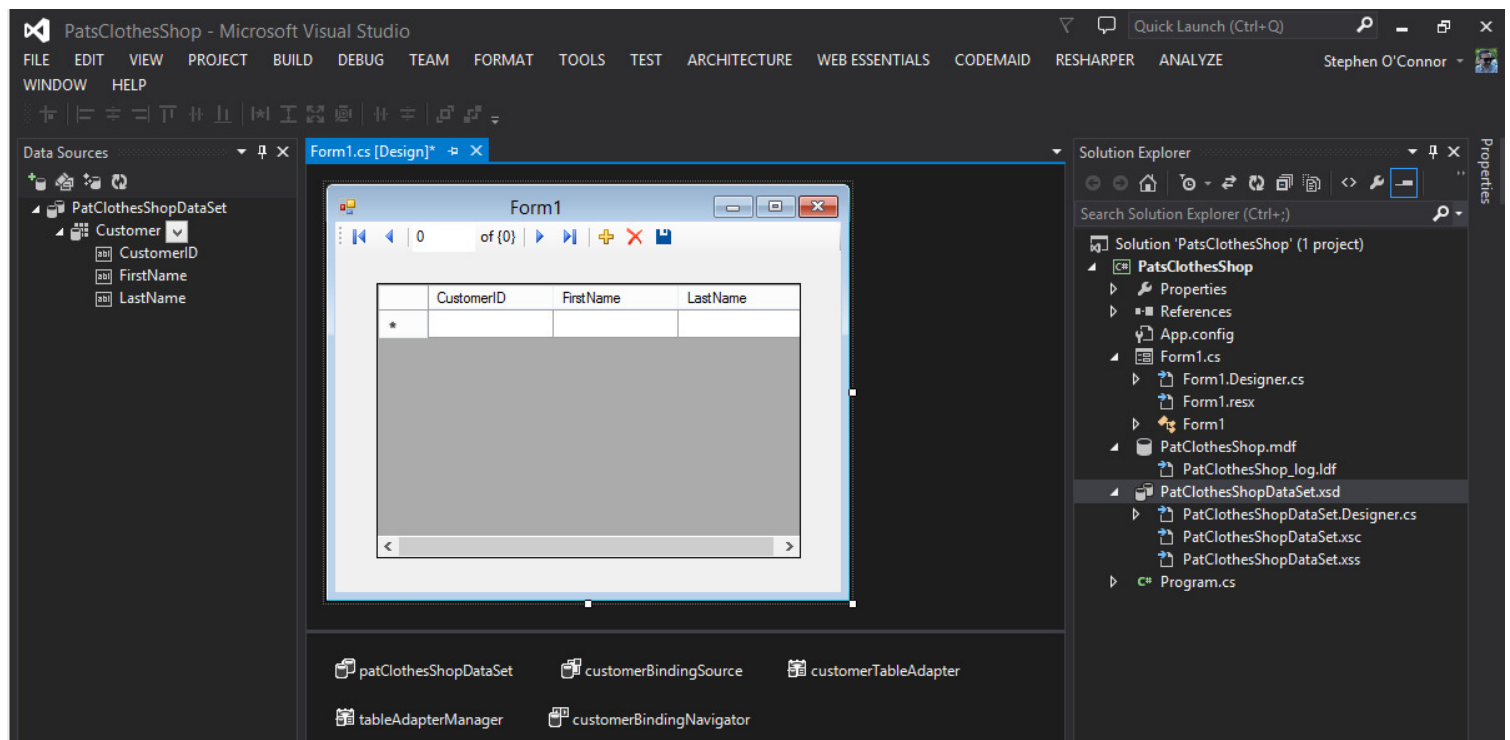
3. select table Customer This creates a .xsd file



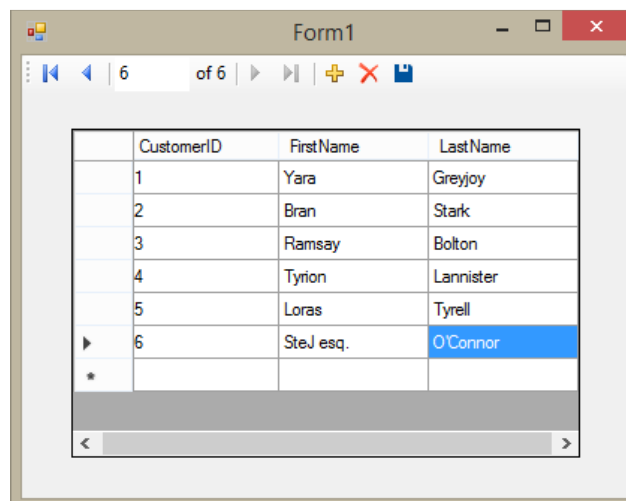
A PatClothesDataSet.xsd file right click on the xsd file and view designer mode. Xsd file is the xml schema document. The xsd file a local copy of database, this file defines the database, temporary stores the data.



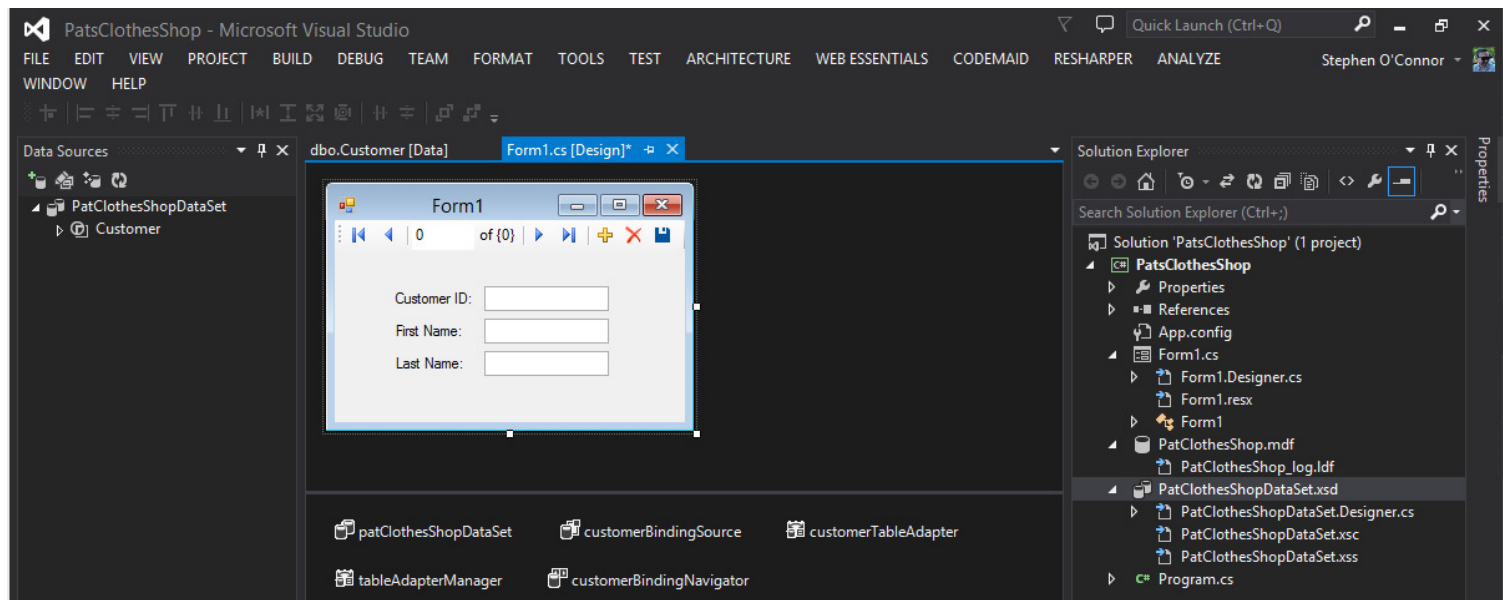
Drag and drop customer table from the Data Sources toolbar.



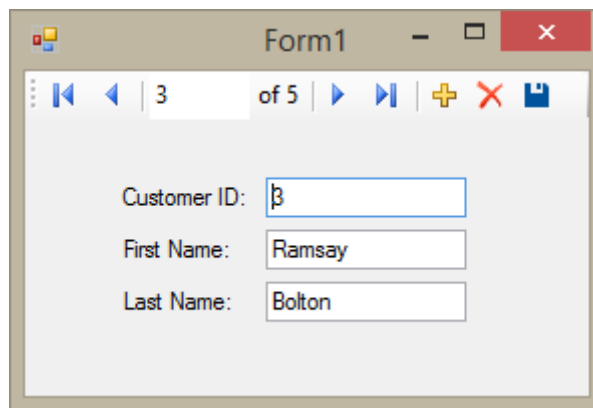
Run the project. A grid of the database navigate through the grid and add an extra row.



To create a Form view; select details from the drop down onto Form1



Run the application and navigate through the details view.



Designer tray.

`patClothesShopDataSet`

The local container for the data within the application. Once the form is opened the dataset gets populated from the data from the database. Temporary storage container.

`patClothesShopBindingSource`

Object /bridge between the information in the dataset and the current row that's being displayed on the form. Keep all of the controls on the form bound to row of data in the DataSet. Co-ordinates what row of data should be currently displayed. The user indicated wanted to go to the first row or the next row or the last row

`patClothesShopTableAdapter`

Retrives data from the database, it contains a connection to the database. Contains an object that connects to the `PatClothesShop.mdf` to retrieve and resolve the information back into the database. Delete, update, add.

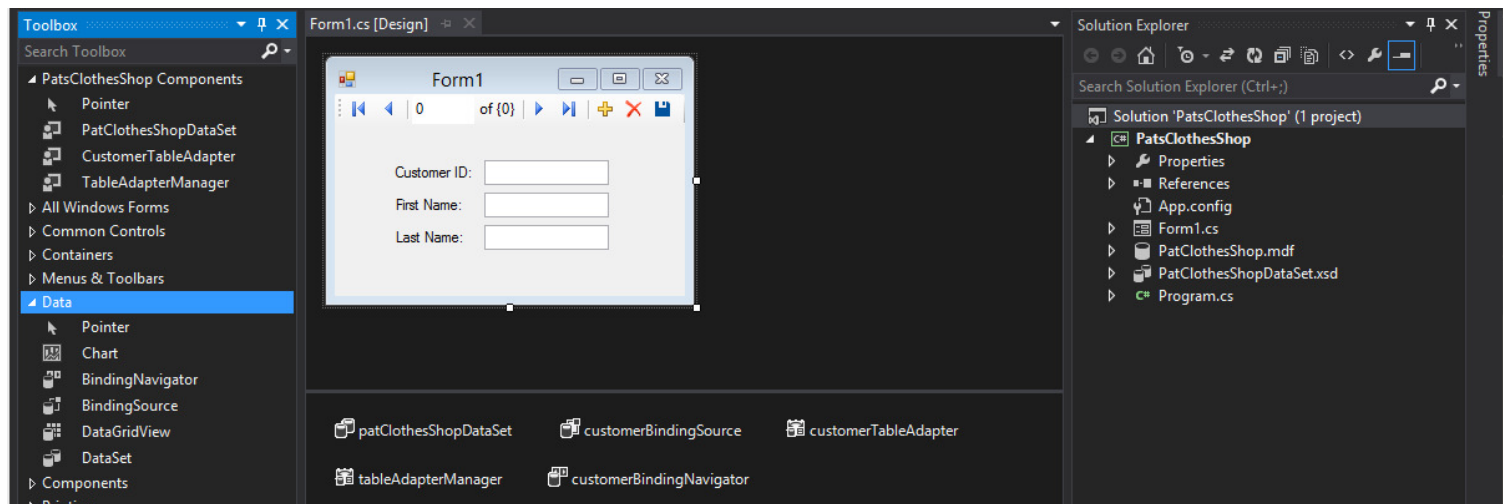
`patClothesShopAdapterManager`

Service interface

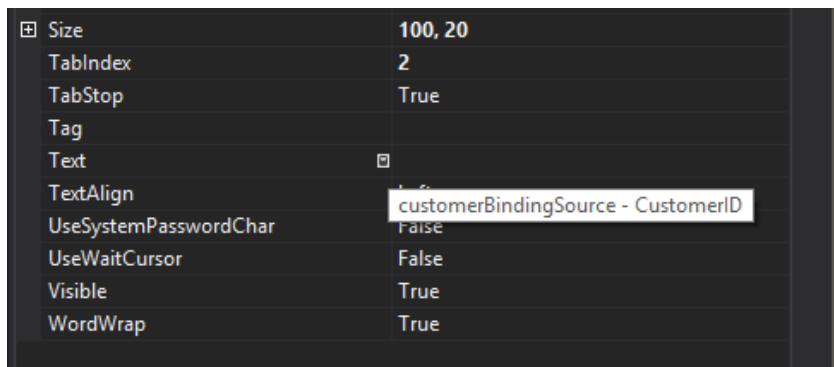
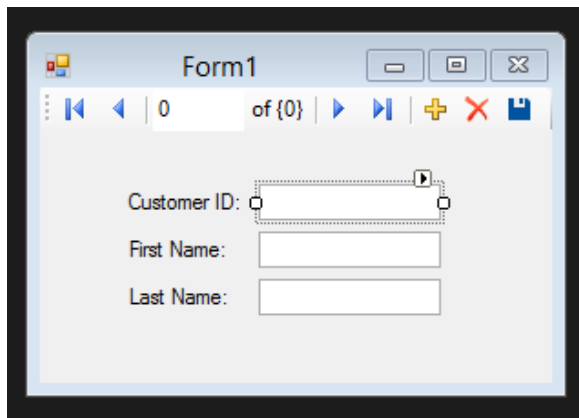
`patClothesShopBindingNavigator`

Toolbar at the top of the form.

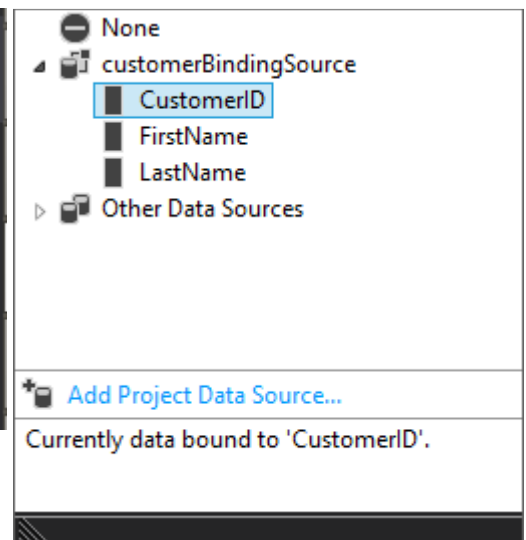
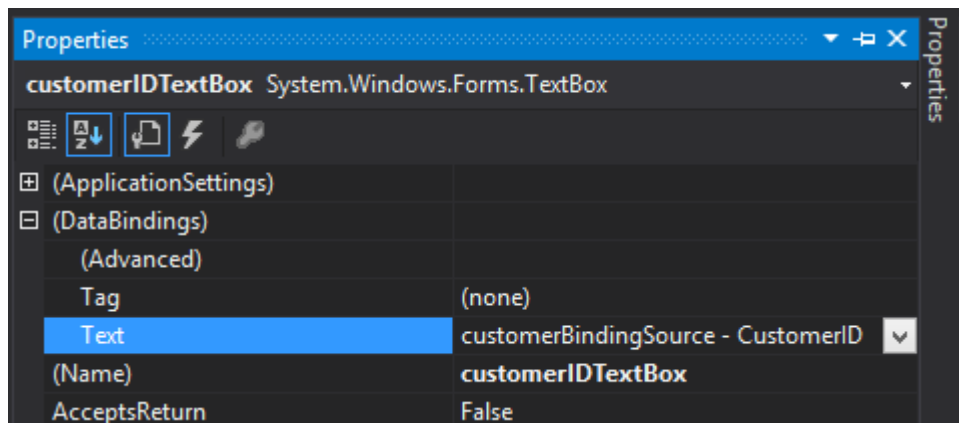
Go to toolbox and select data to show data tools that can be added manually to the form.



Properties of the first textbox. In the Text a Database icon is displayed.



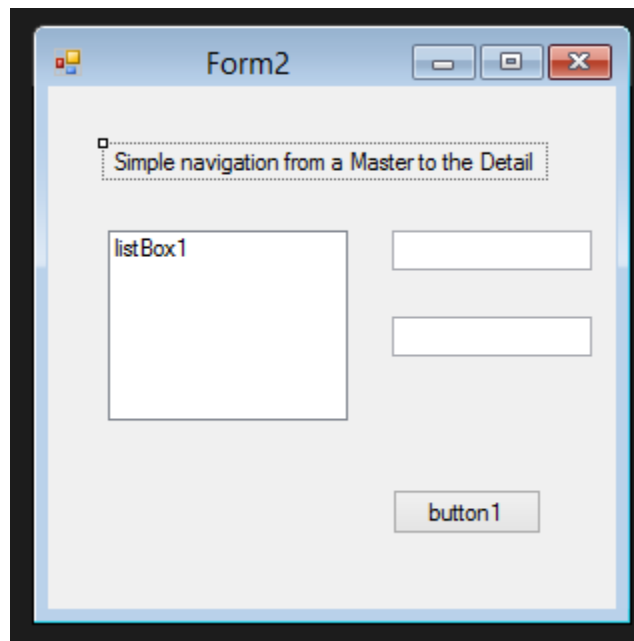
Select and right click to display the properties of the textbox. CustomerID is bound to the first textbox. Binding source schema document. By using the data sources toolbar the database can be dragged and dropped onto the form, sets the textboxes automatically.



Add the data items manually.

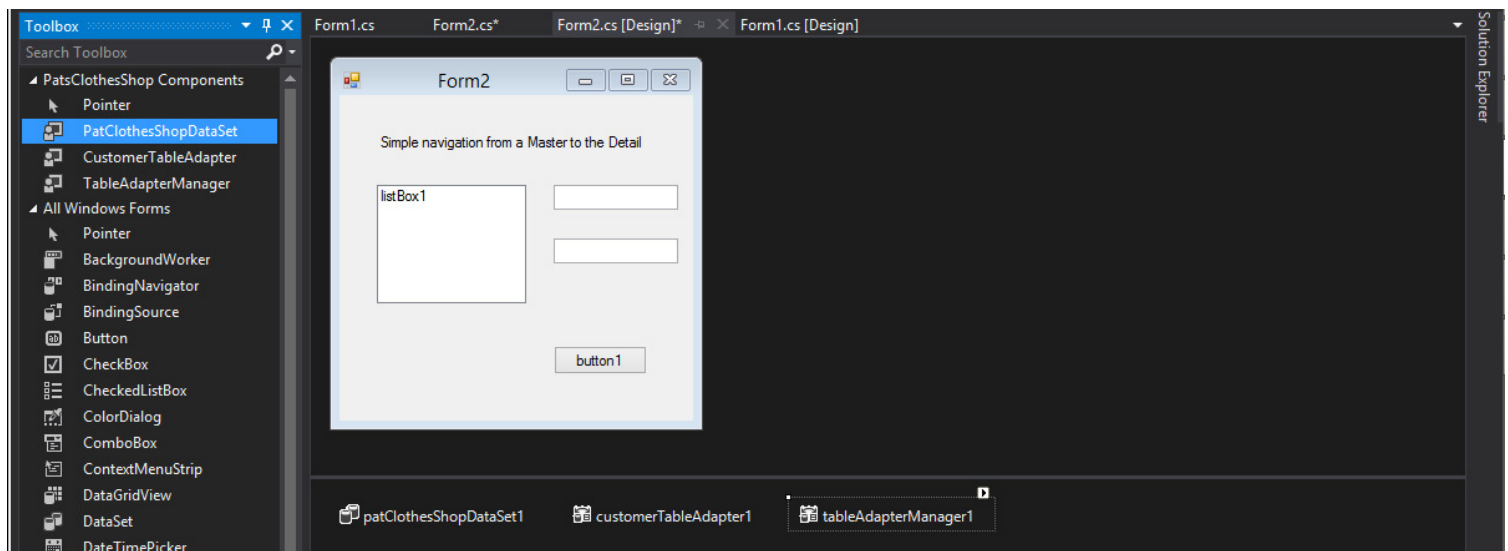
Strongly typed components or preconfigured.

Create a new form form2 add the items displayed in the below image. Label, Listbox, 2 textboxes and a button.



Add a button to form1 to open form2. Type the code below to create an instance of the class Form2

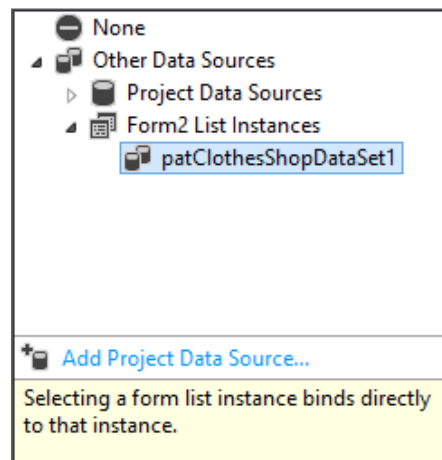
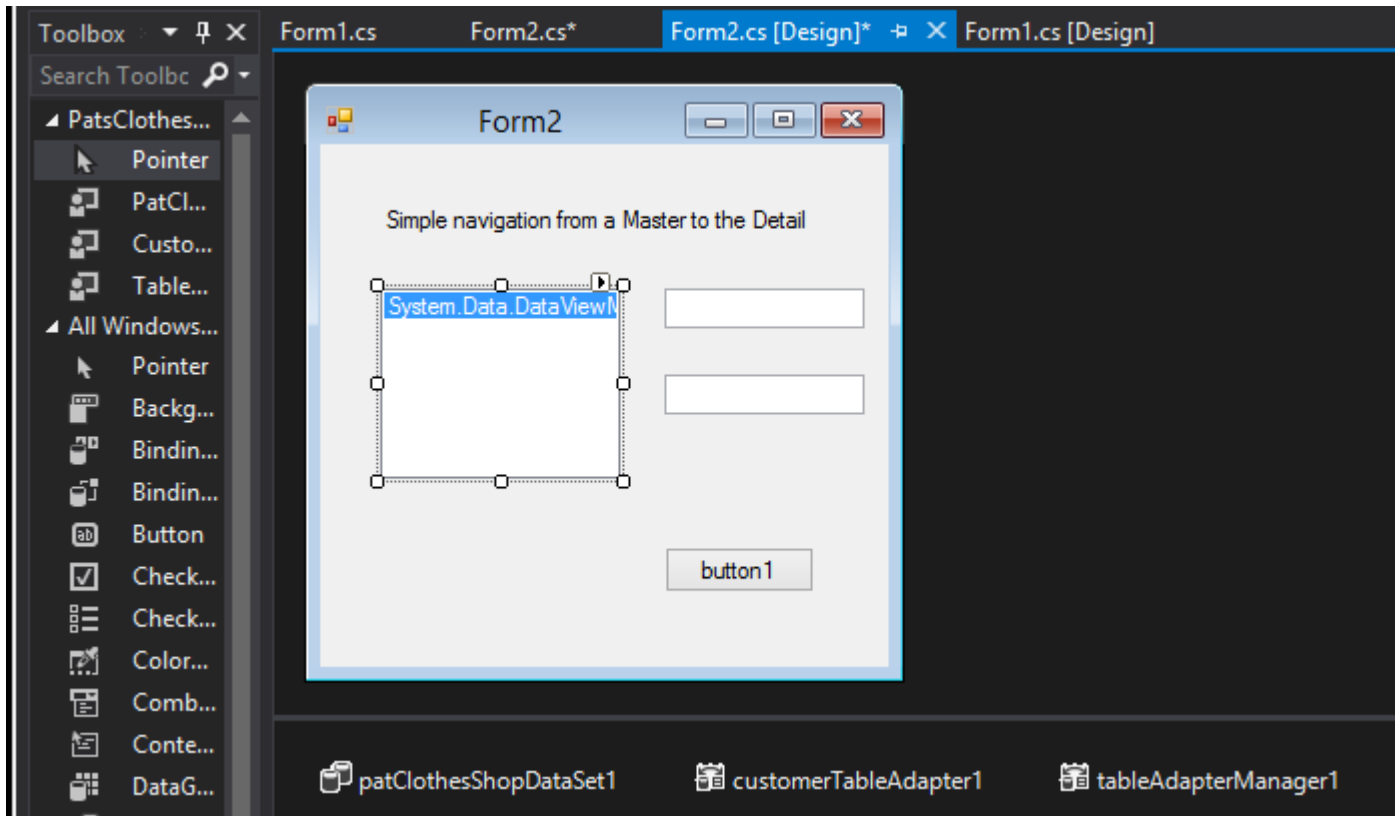
```
Form2 myForm = new Form2();
myForm.Show();
```



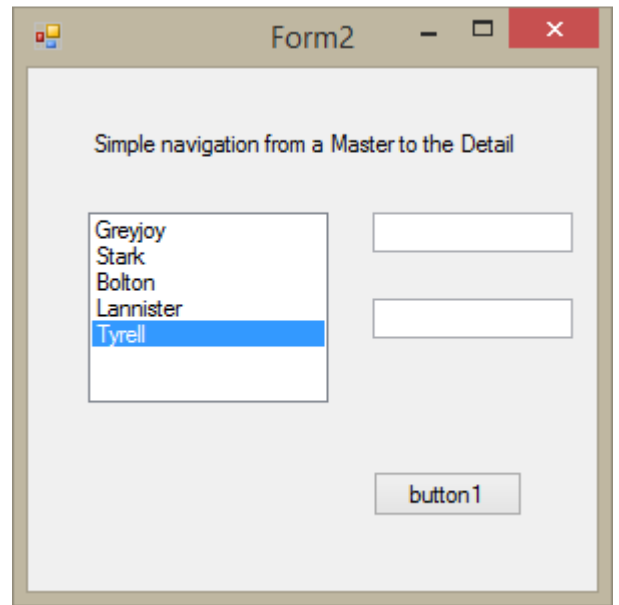
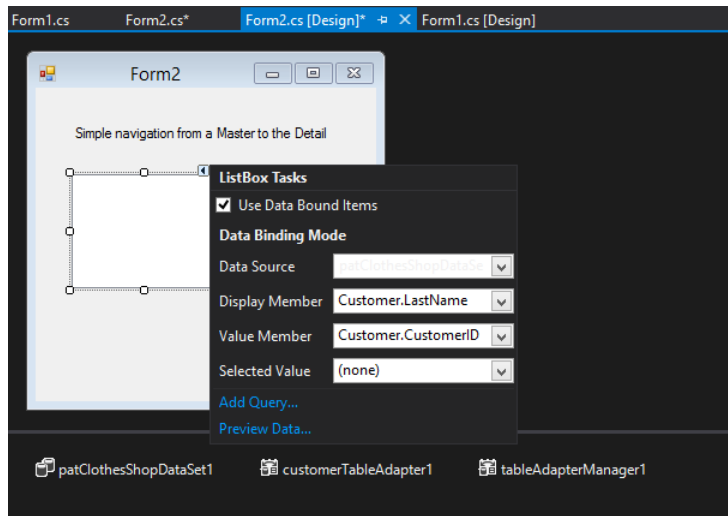
Double click form2 type the code below. Fill method passes in `patClothesShopDataSet1.Customer`. Fill method takes action to grab the data from the database and populate the customer table of the database with the data it retrieves.

```
private void Form2_Load(object sender, EventArgs e)
{
    customerTableAdapter1.Fill(patClothesShopDataSet1.Customer);
}
```

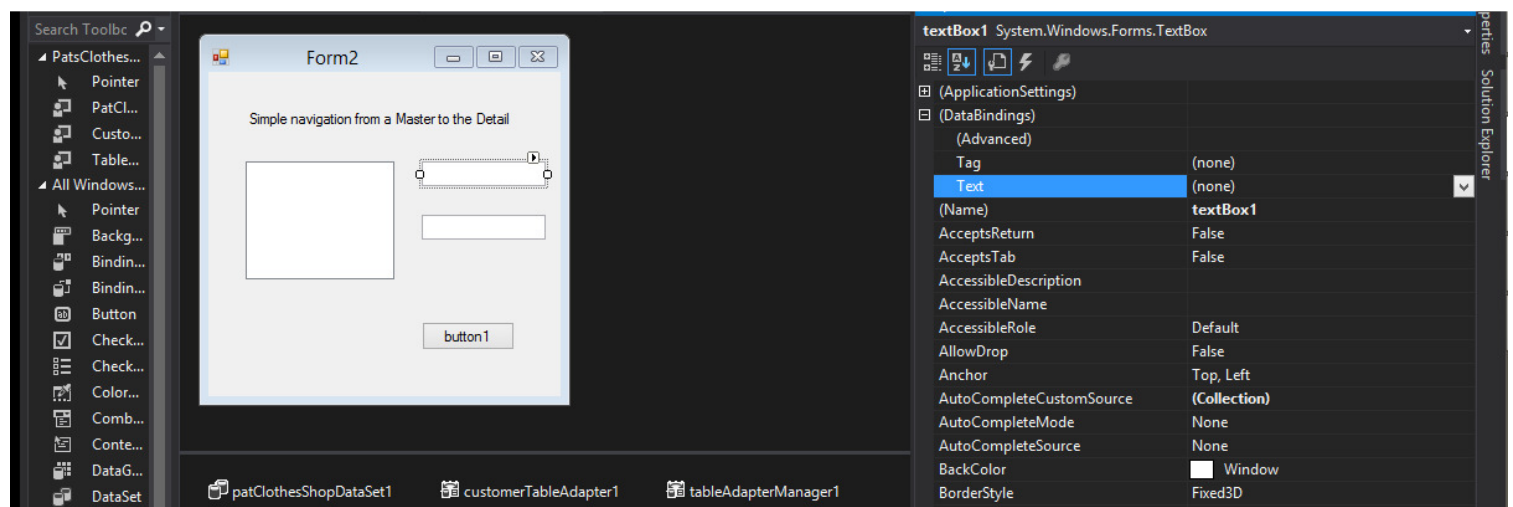
Select the listbox click the arrow and select patClothesShopDataSet1, from the pop-up box.



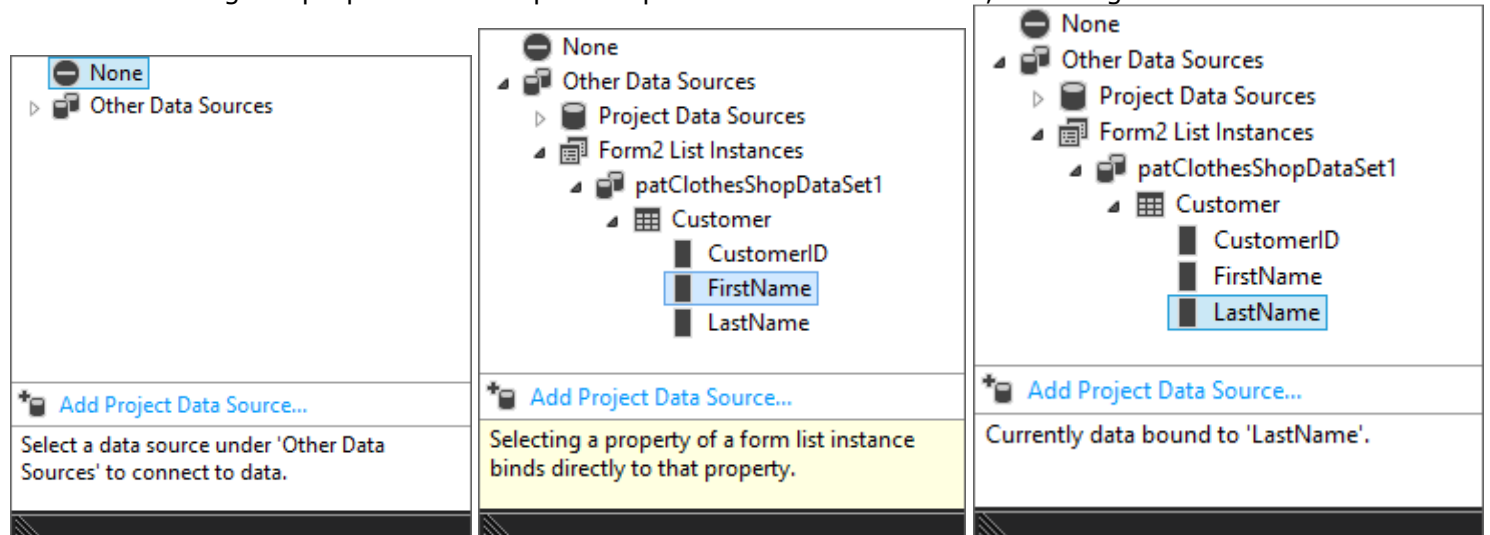
Select the list box and click the arrow button to data bound the listbox to the dataabse. Check the checkbox Use data bound items. From the pop-up select the data source patclothesshop1. In Display member select the last name and Customer last name this will displayed. In value member select the CustomerID, which is always unique.



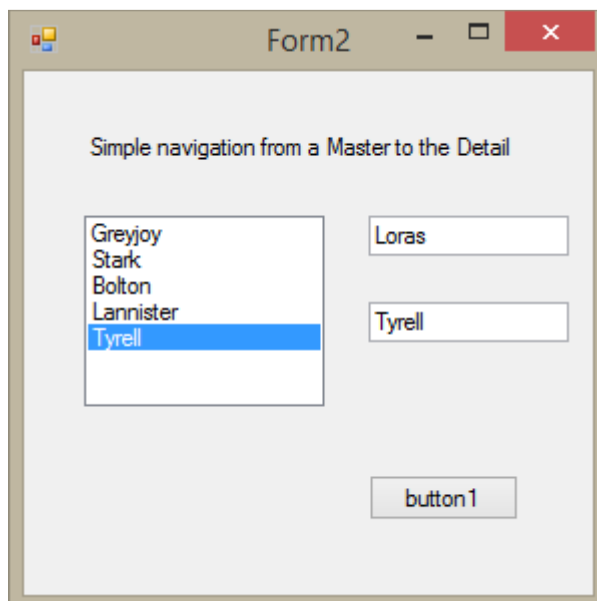
Bind the textboxes on the right with the listbox on the left. Select the first textbox and right click properties go to databindings and select text click the drop down arrow.



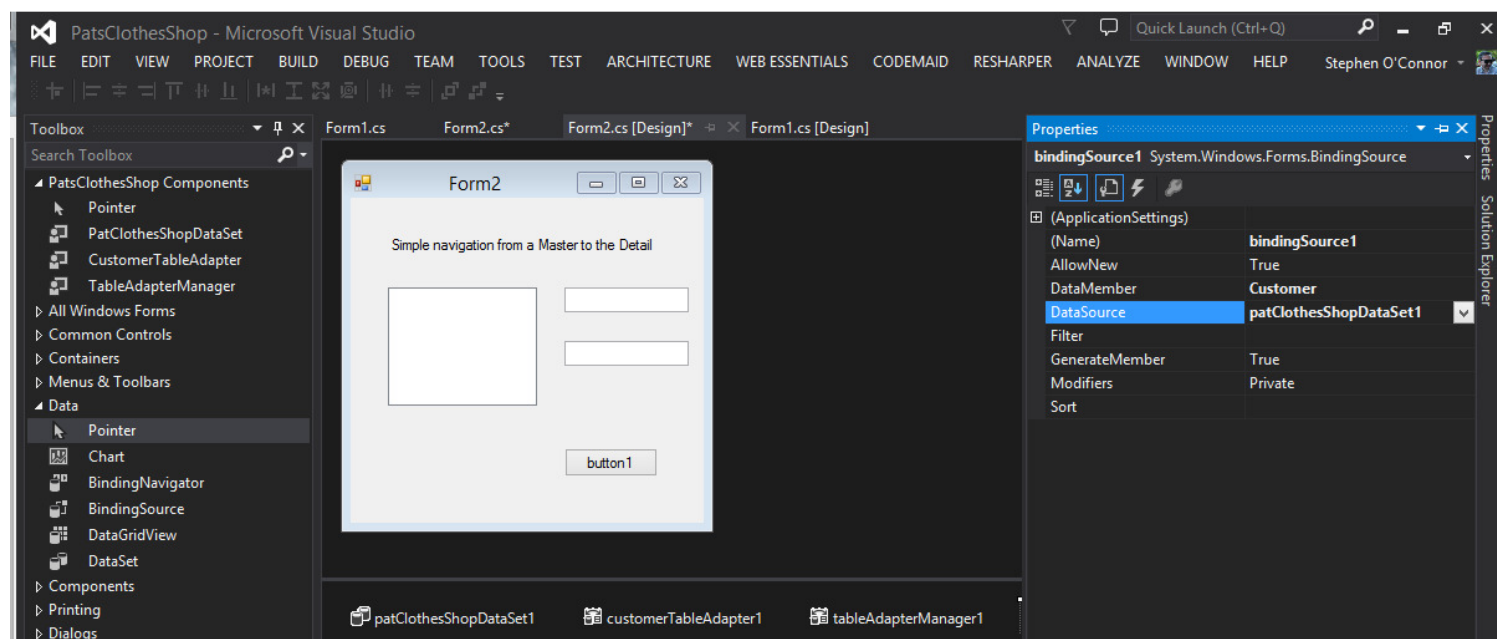
From the pop-up select first name for the first textbox. Go back to the form and select and right click the second textbox go to properties and repeat steps for the second textbox, selecting last name this time.



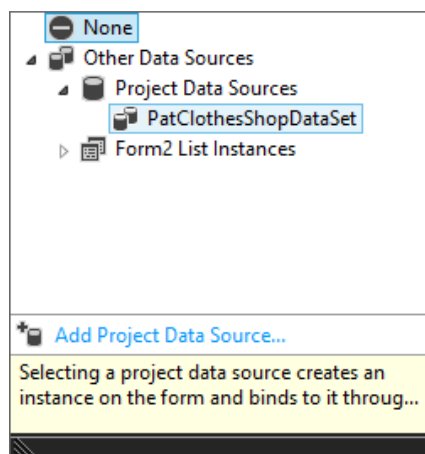
Run the application, open second form. Select from the listbox to display the first name and last name of the person in the textboxes.



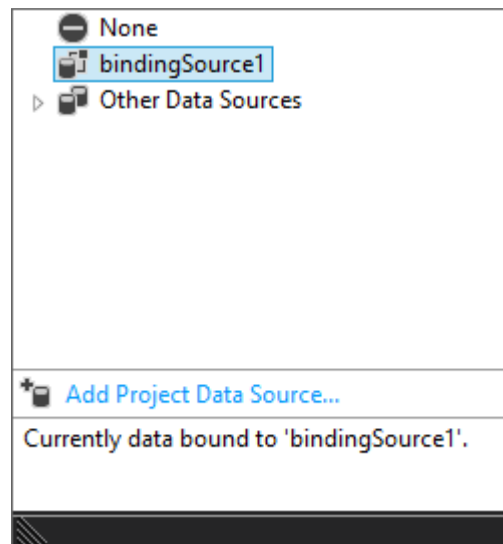
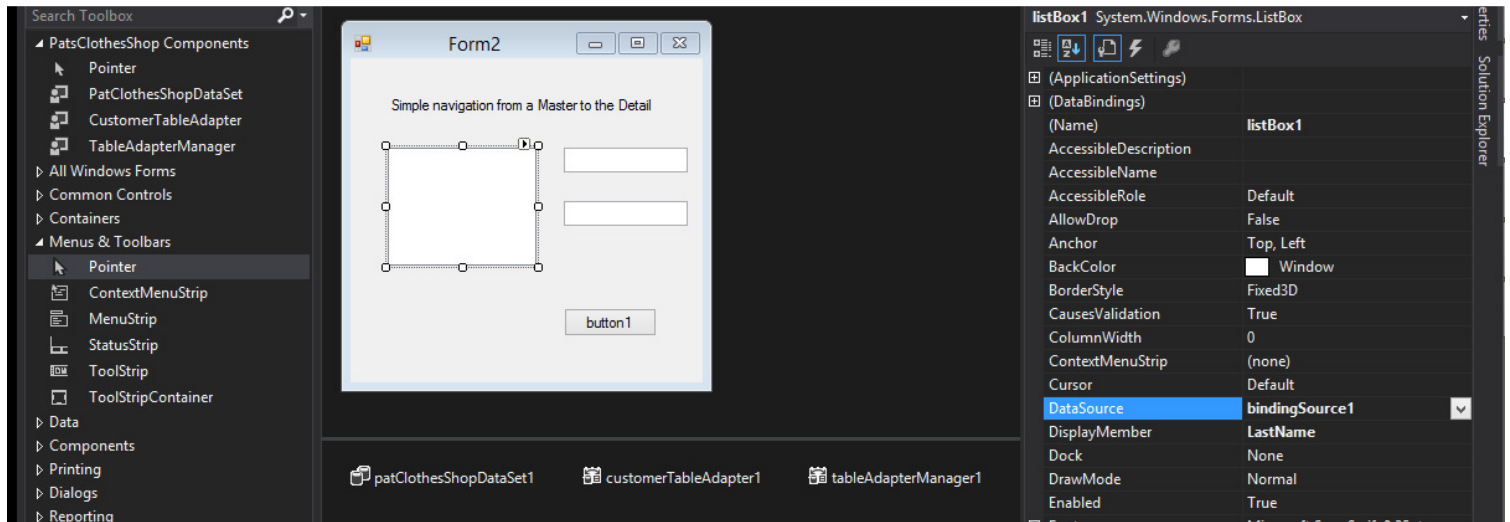
Updating capabilities. Drag and drop the BindingSource onto form2. To update the data. Hover over and read the description of the BindingSource tool. The bindingSource1 has been added to the designer tray.



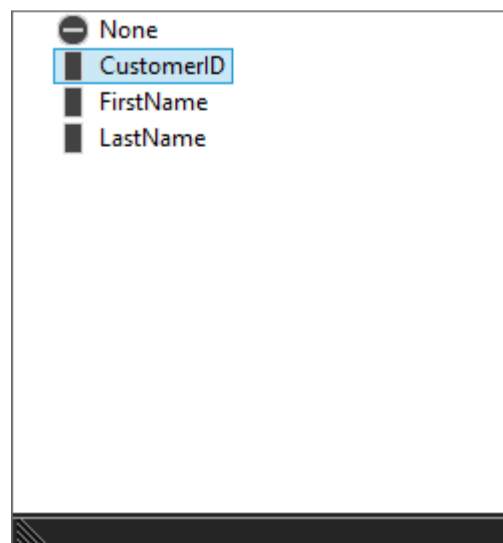
Click bindingSource1 and select PatClothesShopDataSet from the properties DataSource pop-up. Select Customer as the DataMember.

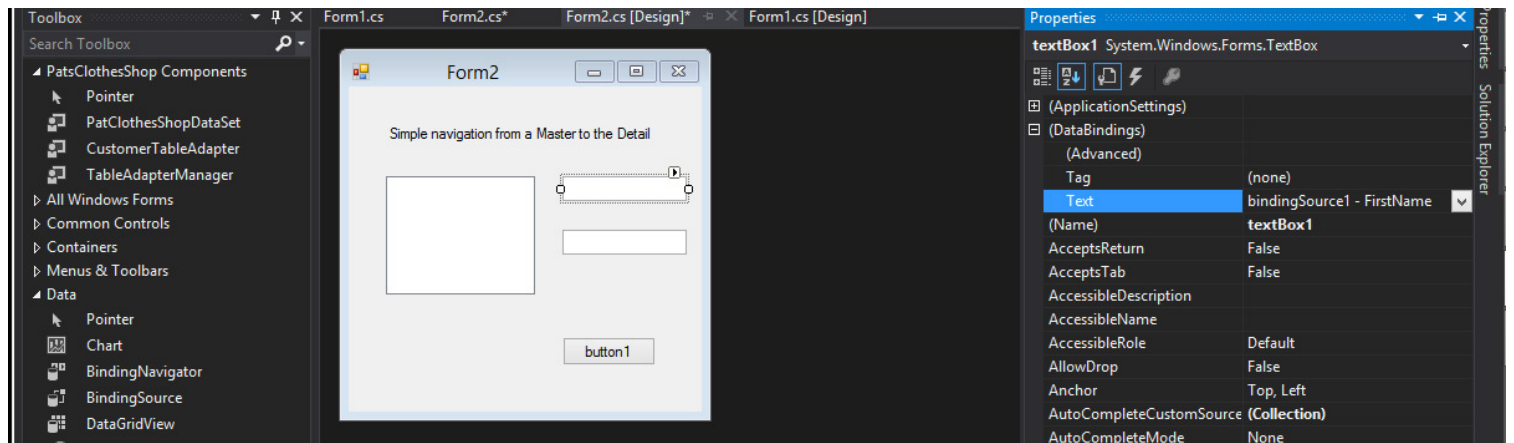


Select listbox go to data source select bindingSource1. Change the DisplayMember to LastName.

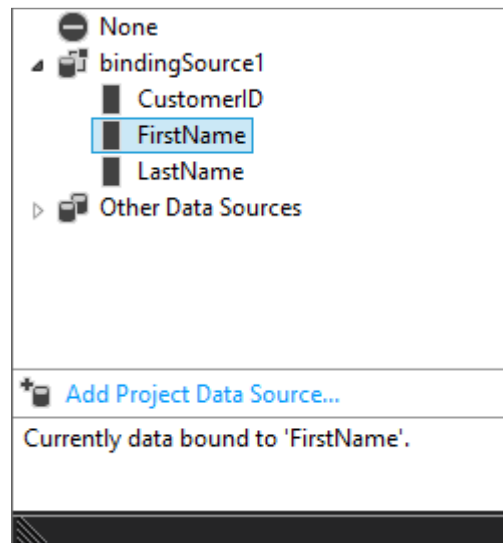


Select Value member in the listbox from properties and change to CustomerID.

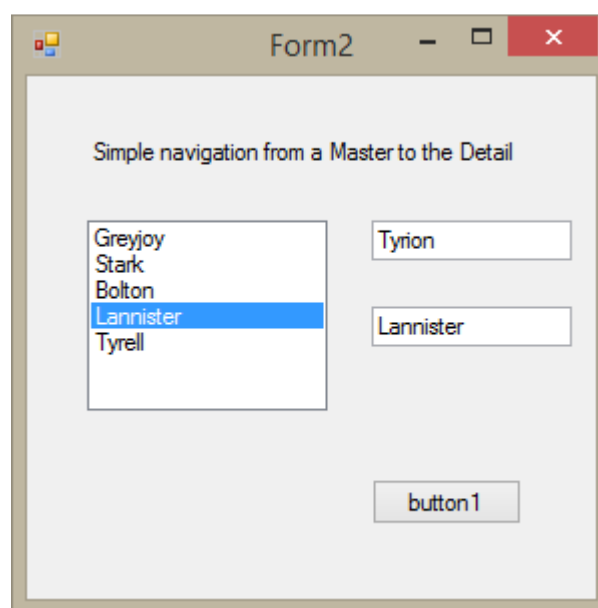




Select the properties of the first and second textboxes and select first and last name respectively.



Run application to check if it still works.



Double click the button on form2

```

18 }
19
20 1 reference | 0 authors | 0 changes
private void button1_Click(object sender, EventArgs e)
21 {
22     bindingSource1.EndEdit();
23     customerTableAdapter1.Update()
24 }
25 ▲ 1 of 6 ▼ int CustomerTableAdapter.Update(DataRow dataRow)
26
27 1 reference | 0 authors | 0 changes
private void Form2_Load(object sender, EventArgs e)
28 {
29     customerTableAdapter1.Fill(patClothesShopDataSet1.Customer);
30 }
31

```

```

private void button1_Click(object sender, EventArgs e)
{
    bindingSource1.EndEdit();
    customerTableAdapter1.Update(patClothesShopDataSet1.Customer);
}

```

```

// save changes to the dataset
bindingSource1.EndEdit();

// select TableAdapter and return number of items updated
customerTableAdapter1.Update(patClothesShopDataSet1.Customer);

```

```

// save changes to the dataset
bindingSource1.EndEdit();

int result = 0;

// return number of items updated
result = customerTableAdapter1.Update(patClothesShopDataSet1.Customer);

// display the row has been updated
MessageBox.Show(result.ToString());

```

Open the .exe file in the debug /bin folder. Change Yara (first name to Theon) close and reopen .exe file and see the change made to the database.

stackoverflow

Actually this is not a issue with update command. Visual Studio keeps two databases. One in project folder and one in bin/debug folder. Database in bin/debug folder always update with Database in project folder. If you view Database through Visual Studio, it always shows the Database inside the project folder not other one inside bin/debug folder.

XML eXtensible Markup Language

Similar to HTML in it's use of the tags <></>

XML contains self-describing structured data but omits presentation (formatting) information.

XML is used to store or transfer data between disparate computing platforms, operating systems, software application, etc.

XML is an open standard, as opposed to a proprietary data format.

Common easily readable format for computer and humans, very portable.

XML used extensively in the .NET Framework.

XML's relationship to ADO.NET Datasets

Allows for a disconnected means of working with data, then allows for easy synchronization of changes back to the original data store.

- XML is used for syndicated RSS feeds
- XML is used to store or transfer data between disparate computing
- XML contains self-describing

Create an XML file called cars.

Create a root element /node `carcollection`, all XML documents have to have a root element.

Inside the `carcollection` element /node create two elements called car these are the sub elements /children of `carcollection`.

Inside the car elements create a sub-element called make.

It can one or the other but not both or it can contain an attribute.

Tag name are created by the programmer.

Pretty much free form, however follow a pattern to make it readable for others and yourself.

A need for conscience valid documents the structure of the XML and follows the rules of XML a well formed XML document.

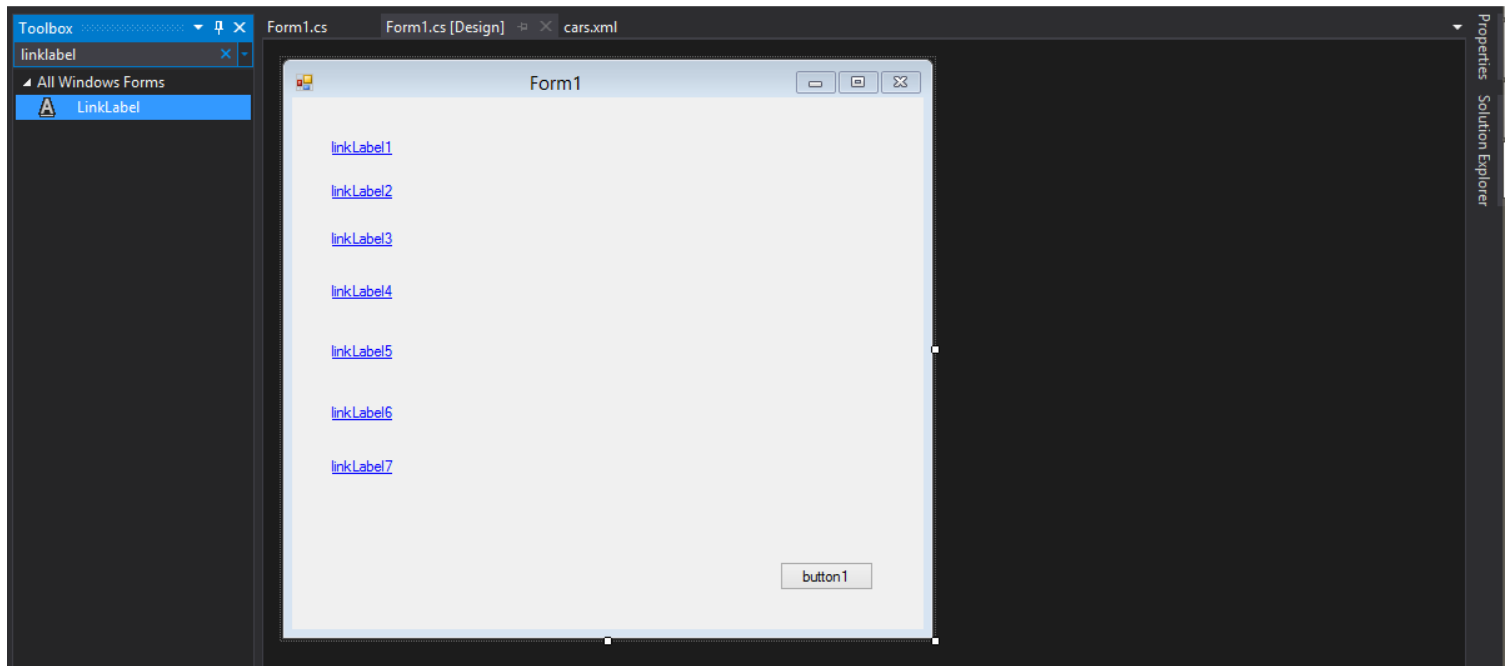
A valid document and well formed.

```
<?xml version="1.0" encoding="utf-8" ?>
<carcollection>
  <car>
    <make>DeLorean</make>
    <model>Time Machine</model>
    <year>1981</year>
  </car>
  <car>
    <make>Cadillac</make>
    <model>Ecto-1</model>
    <year>1984</year>
  </car>
</carcollection>
```

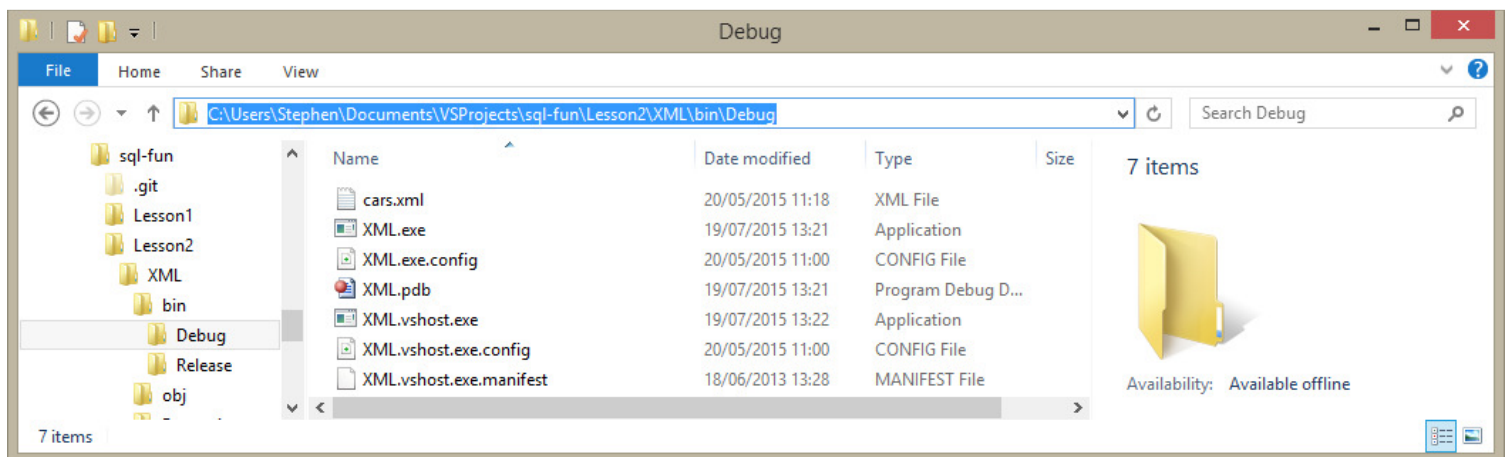
XML Schema Document

Using C# and .NET Framework Class Library to open and navigate through an XML file

Add seven linkLabels and a button



Save the XML document in the bin /debug folder



@ "cars.xml" is the location of the XML file

```
// StreamReader will retrieve the file from the source
// and will convert it into a stream ready to be processed
System.IO.StreamReader sr = new System.IO.StreamReader(@"cars.xml");
```

```
// XmlTextReader
System.Xml.XmlTextReader xr = new System.Xml.XmlTextReader(sr);

// XmlDocument
System.Xml.XmlDocument carCollectionDoc = new System.Xml.XmlDocument();

carCollectionDoc.Load(xr);

// using the InnerText property will give us just the data
// ... since we are at the entire Document level, it will
// give us *all* the values (no delimiter).
linkLabel1.Text = carCollectionDoc.InnerText;
```

My Set-up

The screenshot displays a developer's environment. On the left, a web browser shows the GitHub profile of Stephen J O'Connor (Stevo50). The profile includes a bio, location (Dublin), and a list of repositories. The 'Popular repositories' section lists 'Software-Design-Fundamentals', 'D3-Charts', 'sql-fun', 'StephCakeLiveSearch', and 'Angular-Fundamentals'. The 'Repos contributed to' section lists 'github-ghignone', 'atom/atom', 'marcpalmieri/Caharp-Basics', 'Glavin001/atom-beautify', and 'tapio/live-server'. The 'Contributions' section shows a heatmap of activity over time, with a summary of 1,117 total contributions, 184 days of activity, and a current streak of 0 days. The 'Contribution activity' section shows 12 commits over the last week.

On the right, a PowerShell terminal window shows the following commands and output:

```
You are now entering PowerShell : Stephen
PS C:\Users\Stephen> cd C:\Users\Stephen\Documents\VSProjects\sql-fun
sql-fun master $ git add .
sql-fun master $ git commit -am 'Lesson Plan clean up'
[master b1489aa] Lesson Plan clean up
1 file changed, 0 insertions(+), 0 deletions(-)
sql-fun master $ git push origin master
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), 269.37 KiB | 0 bytes/s, done.
Writing objects: 100% (3/3), 269.37 KiB | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
To https://github.com/Stevo50/sql-fun.git
9e0d417..b1489aa master -> master
sql-fun master $
```