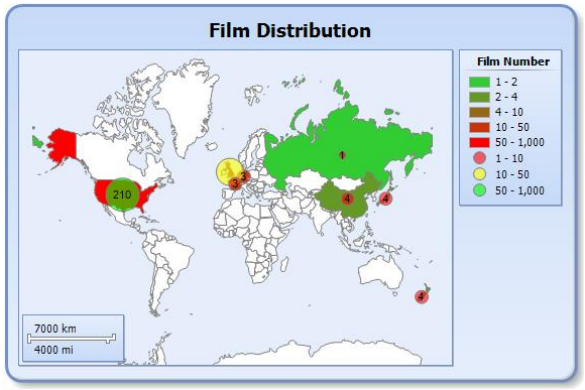
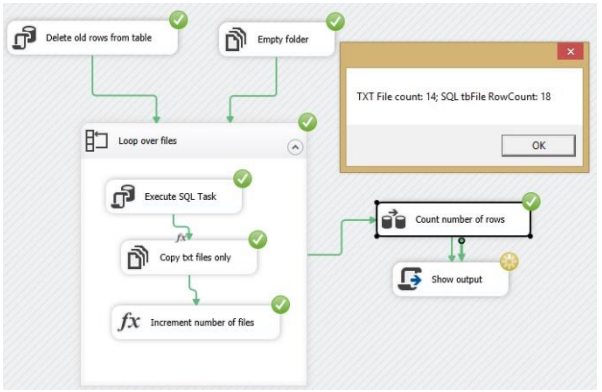
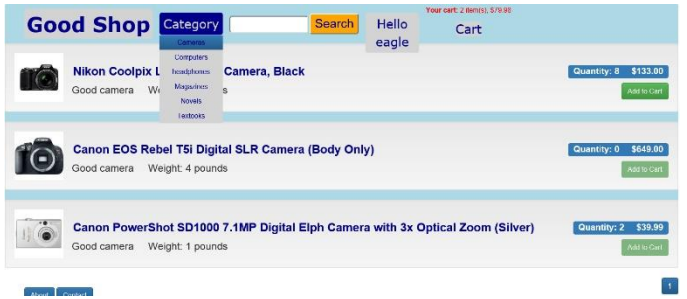
SQL Server and Programming Demos



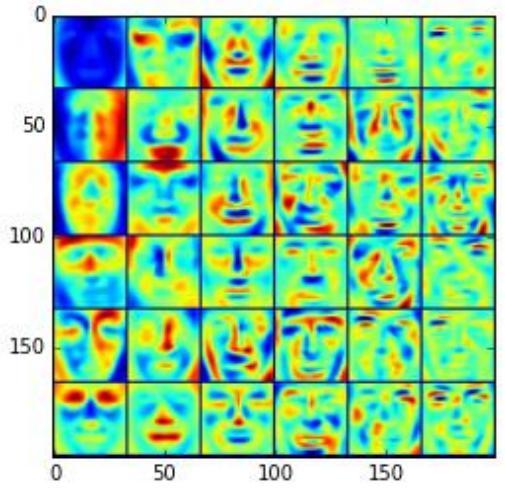
SSRS



SSIS



Website



Python

Qi Charles Sun

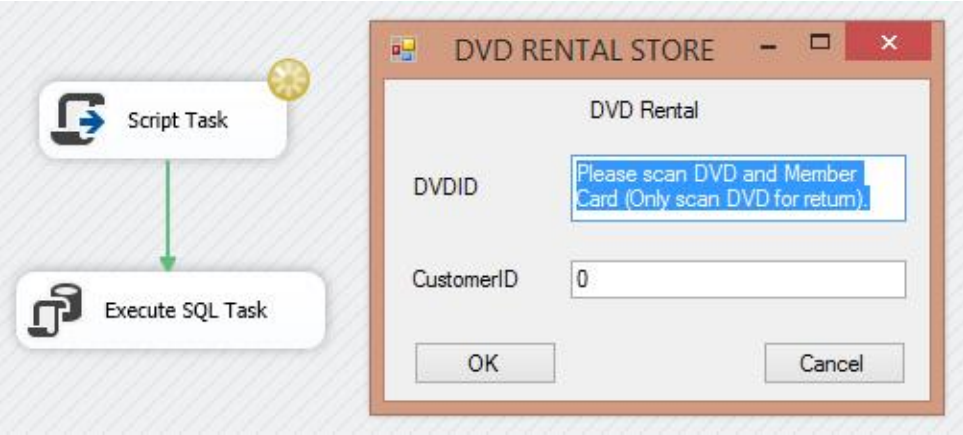
Outline

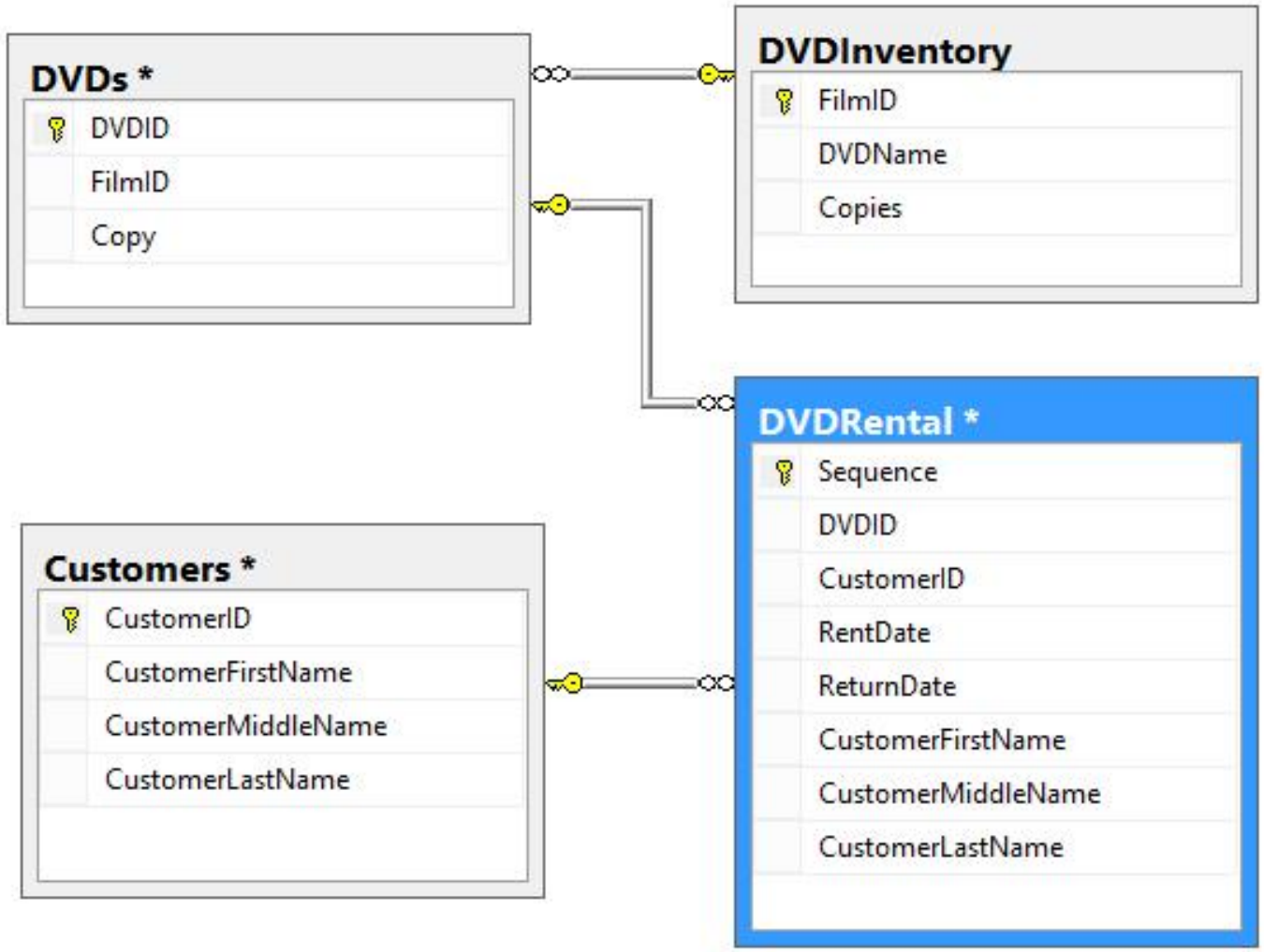
1. Database demos using T-SQL, SSIS, C#
2. Website demos
3. SSRS demos
4. SSIS demos
5. SSAS demos
6. Excel BI demos
7. Programing demos (Python, Matlab, C#, and Labview)

Objective of Project (database1)

1. Build a OLTP database for DVD rental
2. Created 4 tables with Primary Key and Foreign Key constraints
3. Recorded every transaction for return and rental
4. Input the new member to Table Customers
5. Minimized inputs: scan of DVD and member card for rental, and scan of DVD for return
6. Two triggers and one stored procedure to accomplish the goal
7. SSIS and C# generated two prompt windows for inputs

Database diagram and prompt input window

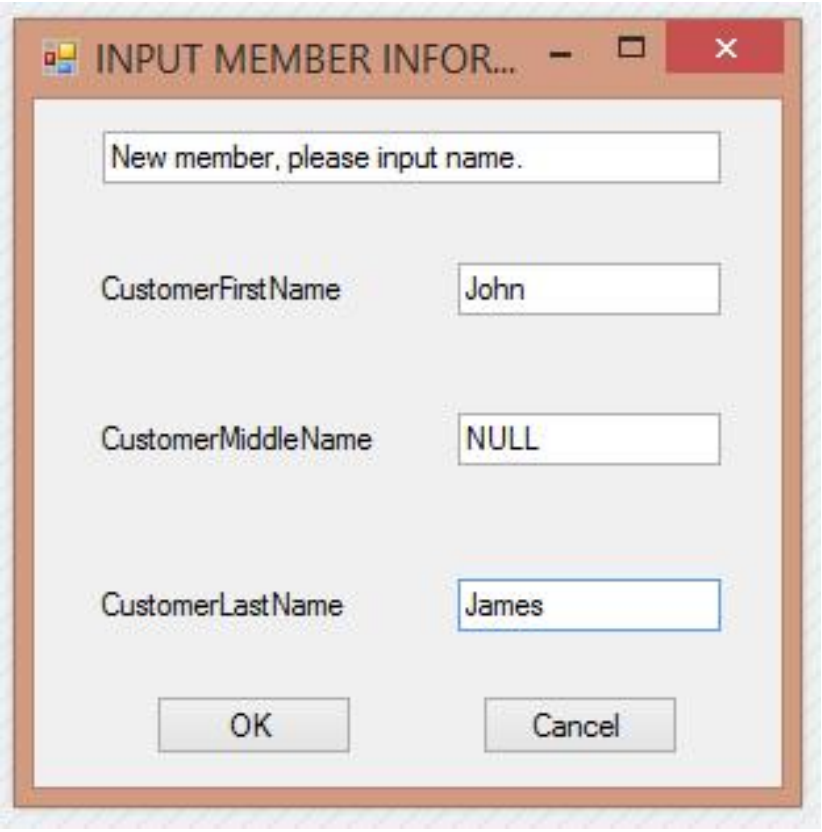


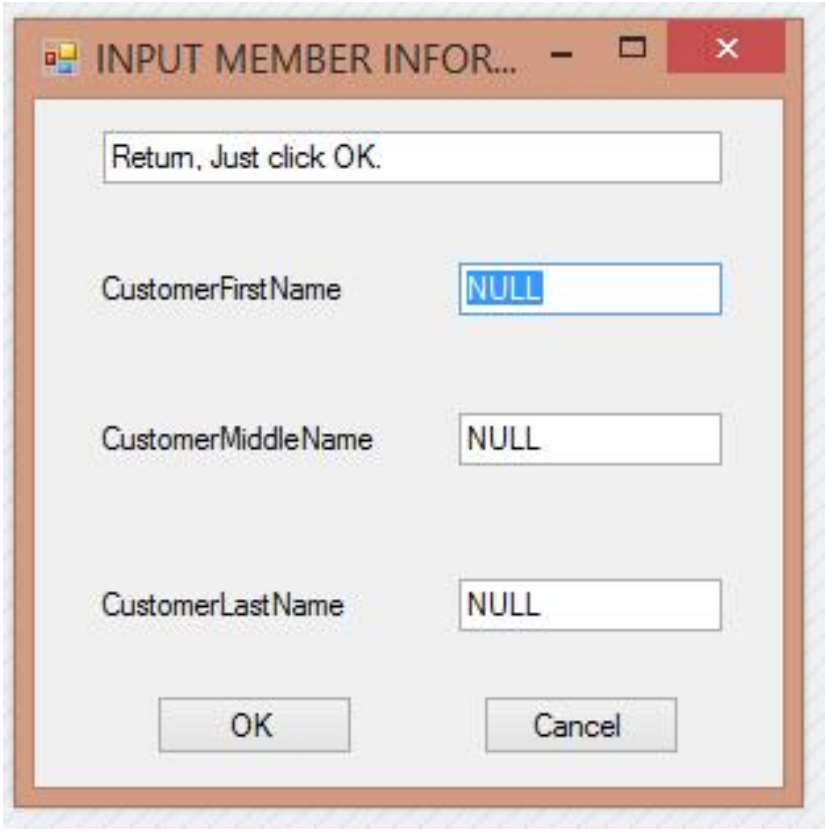


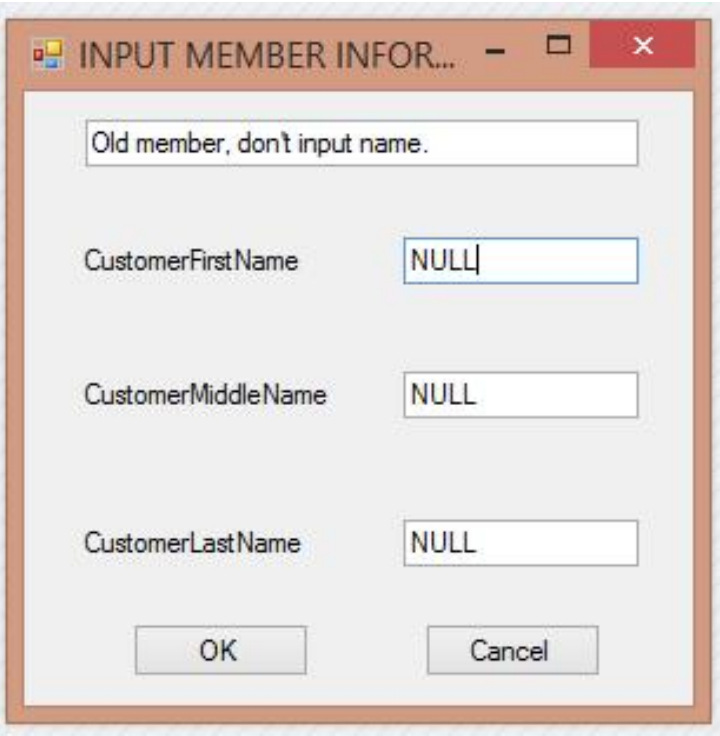
4 tables connected by primary key and foreign key constraints. Tables can contain more columns for details such as DVD condition, member address. The database has its basic functions. The input window was designed using two windows forms by C#.

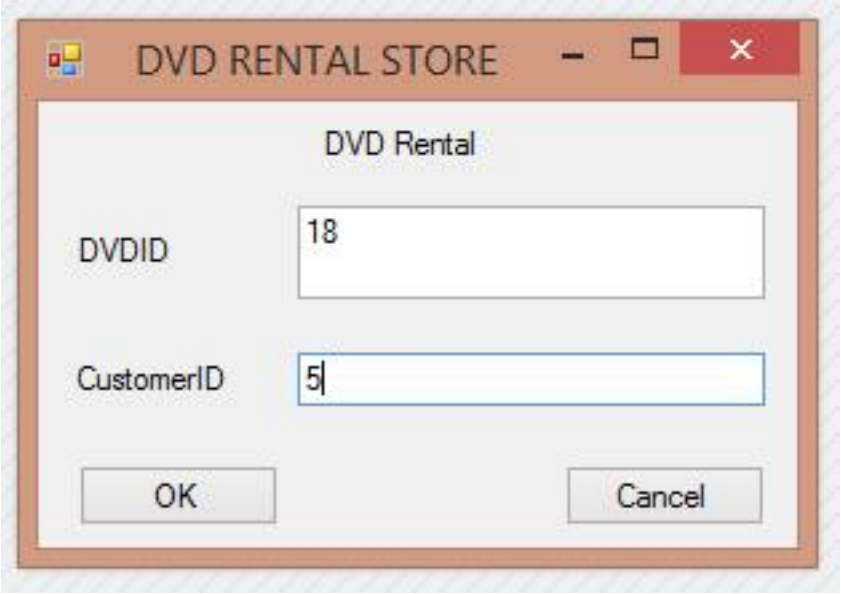
https://github.com/charleseagle/DVD-Rental-Database-Design-SSIS-Cshape

Different functionalities









https://github.com/charleseagle/DVD-Rental-Database-Design-SSIS-Cshape

Two triggers

```
CREATE TRIGGER Rent ON DVDRental
AFTER INSERT
AS
BEGIN
    UPDATE DVDInventory
    SET DVDInventory.Copies =
    (SELECT DVDInventory.Copies - 1
    FROM DVDs INNER JOIN inserted
    ON DVDs.DVDID = inserted.DVDID
    INNER JOIN DVDInventory ON DVDInventory.FilmID = DVDs
    WHERE DVDInventory.FilmID = (SELECT FilmID FROM inser
                                 INNER JOIN DVDs ON DV
    UPDATE DVDS
    SET Copy = 0
    WHERE DVDID = (SELECT DVDID FROM inserted)
END;
GO
```

```
CREATE TRIGGER DVDReturn ON DVDRental
AFTER UPDATE
AS
BEGIN
    UPDATE DVDInventory
    SET DVDInventory.Copies =
    (SELECT DVDInventory.Copies + 1
    FROM DVDInventory AS a INNER JOIN DVDs AS b ON a.F
    INNER JOIN inserted AS c ON c.DVDID = b.DVDID)
    WHERE DVDInventory.FilmID = (SELECT FilmID FROM DV
                                 ON a.DVDID = b.DVDID)
    UPDATE DVDs
    SET Copy = 1
    WHERE DVDID = (SELECT DVDID FROM inserted)
END;
GO
```

https://github.com/charleseagle/DVD-Rental-Database-Design-SSIS-Cshape

One stored procedure

Used IF clause to finish different INSERT and UPDATE, and remind the DVD is out of stock.

```
CREATE PROCEDURE dbo.uspRental @DVDID int,
            @CustomerID int,
            --@ReturnDate datetime = NULL,
            @CustomerFirstName nvarchar(255) = NULL,
            @CustomerMiddleName nvarchar(255) = NULL,
            @CustomerLastName nvarchar(255) = NULL
AS
BEGIN
    SET NOCOUNT ON;
    IF (SELECT Copy FROM DVDs WHERE DVDID = @DVDID) = 1
    BEGIN
        IF @CustomerID IN (SELECT CustomerID FROM Customers)
        BEGIN
            INSERT INTO DVDRental (DVDID, CustomerID, RentDate, CustomerFirstName, CustomerMiddleName, CustomerLastName)
            VALUES(@DVDID, @CustomerID, GETDATE(),
            (SELECT CustomerFirstName FROM Customers WHERE CustomerID = @CustomerID),
            (SELECT CustomerMiddleName FROM Customers WHERE CustomerID = @CustomerID),
            (SELECT CustomerLastName FROM Customers WHERE CustomerID = @CustomerID))
        END
        ELSE
        BEGIN
            SET NOCOUNT ON;
            INSERT INTO Customers
            VALUES(@CustomerID, @CustomerFirstName, @CustomerMiddleName, @CustomerLastName)
            INSERT INTO DVDRental (DVDID, CustomerID, RentDate, CustomerFirstName, CustomerMiddleName, CustomerLastName)
            VALUES (@DVDID, @CustomerID, GETDATE(), @CustomerFirstName, @CustomerMiddleName, @CustomerLastName)
        END
    END
    ELSE
    BEGIN
        UPDATE DVDRental
        SET ReturnDate = GETDATE(), DVDID = @DVDID
        WHERE Sequence = (SELECT MAX(Sequence) FROM DVDRental)
    END
IF (SELECT Copies FROM DVDInventory AS a INNER JOIN DVDs AS b ON a.FilmID = b.FilmID WHERE b.DVDID = @DVDID) = 0
    PRINT 'OUT OF STOCK'
END;
GO
```

https://github.com/charleseagle/DVD-Rental-Database-Design-SSIS-Cshape

C# code demo

Used two windows forms to accomplished the goal. The only input is scan of DVD and member car for rental, and scan of DVD for return. If the member is new, the system can record his/her information.

```
public Form2()
{
    InitializeComponent();
}
1 reference
private void Form2_Load(object sender, EventArgs e)
{
    CustomerFirstName.Text = "NULL";
    CustomerMiddleName.Text = "NULL";
    CustomerLastName.Text = "NULL";

    SqlConnection myConnection = new SqlConnection("server = eagle;" + "Trusted_Connection = yes;"
            + "database = DVDLibrary;" + "Connection timeout = 30");
    myConnection.Open();
    SqlCommand myCommand = new SqlCommand("SELECT CustomerID FROM Customers", myConnection);
    SqlDataReader NewOrOld = myCommand.ExecuteReader();
    ScriptMain IDs = new ScriptMain();
    bool Showed = true;
    var b = IDs.FirstForm(Showed);
    int a = b.Item2;

    while (NewOrOld.Read())
    {
        if (NewOrOld.GetValue(0).ToString() != Convert.ToString(a))
        {
            Indicator.Text = "New member, please input name.";

        }
        else
        {
            Indicator.Text = "Old member, don't input name.";
            break;
        }

    }
    if (a == 0)
    {
        Indicator.Text = "Return, Just click OK.";
    }
```

Objective of Project (database2)

1. Build a database with the function to store error information
2. Created tables with setting Primary Key and Non-Clustered Index
3. Loaded all frequency range spectroscopic data to the database using “BULK INSERT” or “bcp” in PowerShell
4. Using SQL commends to calculate and correct data, and saved the results to new tables
5. Merged the data using “UNION” commend for the FIR range data, and saved the result to the created table
6. Merged the shifted and shortened data using “UNION ALL” for all frequency range data, and saved the result to the created table

Tables: All frequency range tables and merged tables saved to the schema [2H]

Design Diagram

Load data (BULK INSERT or bcp)

Merge and manipulate data using SQL Server

Export data (SSMS or bcp)

Create Database

- Creating a database that specifies the data and transaction log files
- Alter the database SET options:

RECOVERY SIMPLE: A simple backup strategy that uses minimal log
ANSI_NULLS: All comparisons to a null value evaluate to UNKNOWN
ANSI_PADDING: Strings are padded to the same length
ANSI_WARNINGS: Errors or warnings are issued (divide-by-zero or null values in aggregate functions.
ARITHABORT: A query is ended (an overflow or divide-by-zero error)
CONCAT_NULL_YIELDS_NULL: The result of a concatenation operation is NULL when either operand is NULL
QUOTED_IDENTIFIER: Double quotation marks can be used to enclose delimited identifiers.
NUMERIC_ROUNDABORT: An error is generated when loss of precision occurs in an expression.
PAGE_VERIFY CHECKSUM: Discovers damaged database pages caused by disk I/O path errors
ALLOW_SNAPSHOT_ISOLATION: Turns off the Snapshot option at the database level

```
CREATE DATABASE [Spectroscopy]
    ON (NAME = 'Spectroscopy_Data', FILENAME =
N'E:\SQL_Server_Practice\Spectroscopy_Data.mdf'
, SIZE = 5, FILEGROWTH = 8) -- SIZE = 120,
    LOG ON (NAME = 'Spectroscopy_Log', FILENAME
=
N'E:\SQL_Server_Practice\Spectroscopy_Log.ldf'
, SIZE = 2, FILEGROWTH = 96);
GO

ALTER DATABASE [Spectroscopy]
SET RECOVERY SIMPLE,
    ANSI_NULLS ON,
    ANSI_PADDING ON,
    ANSI_WARNINGS ON,
    ARITHABORT ON,
    CONCAT_NULL_YIELDS_NULL ON,
    QUOTED_IDENTIFIER ON,
    NUMERIC_ROUNDABORT OFF,
    PAGE_VERIFY CHECKSUM,
    ALLOW_SNAPSHOT_ISOLATION OFF;
GO
```


https://github.com/charleseagle/Database-for-merging-and-correcting-data-SQL-Server

Create table to store error information

ON [PRIMARY] will create the structures on the "Primary“ filegroup

Alter the table to add a clustered index

Create a stored procedure

uspPrintError prints error information about the error that caused execution to jump to the CATCH block of a TRY...CATCH construct. Should be executed from within the scope of a CATCH block otherwise it will return without printing any error information.

```
CREATE TABLE [dbo].[ErrorLog](
    [ErrorLogID] [int] IDENTITY (1, 1) NOT NULL,
    [ErrorTime] [datetime] NOT NULL
    CONSTRAINT [DF_ErrorLog_ErrorTime] DEFAULT (GETDATE()),
    [UserName] [sysname] NOT NULL,
    [ErrorNumber] [int] NOT NULL,
    [ErrorSeverity] [int] NULL,
    [ErrorState] [int] NULL,
    [ErrorProcedure] [nvarchar](126) NULL,
    [ErrorLine] [int] NULL,
    [ErrorMessage] [nvarchar](4000) NOT NULL
) ON [PRIMARY];
GO

ALTER TABLE [dbo].[ErrorLog] WITH CHECK ADD
    CONSTRAINT [PK_ErrorLog_ErrorLogID] PRIMARY KEY CLUSTERED
    (
        [ErrorLogID]
    )  ON [PRIMARY];
GO
CREATE PROCEDURE [dbo].[uspPrintError]
AS
BEGIN
    SET NOCOUNT ON;

    -- Print error information.
    PRINT 'Error ' + CONVERT(varchar(50), ERROR_NUMBER()) +
          ', Severity ' + CONVERT(varchar(5), ERROR_SEVERITY()) +
          ', State ' + CONVERT(varchar(5), ERROR_STATE()) +
          ', Procedure ' + ISNULL(ERROR_PROCEDURE(), '-') +
          ', Line ' + CONVERT(varchar(5), ERROR_LINE());
    PRINT ERROR_MESSAGE();
END;
GO
```


https://github.com/charleseagle/Database-for-merging-and-correcting-data-SQL-Server

Create a stored procedure

uspLogError logs error information in the ErrorLog table about the error that caused execution to jump to the CATCH block of a TRY...CATCH construct. This should be executed from within the scope of a CATCH block otherwise it will return without inserting error information.

```
CREATE PROCEDURE [dbo].[uspLogError]
    @ErrorLogID [int] = 0 OUTPUT -- contains the ErrorLogID of the row inserted
AS                               -- by uspLogError in the ErrorLog table
BEGIN
    SET NOCOUNT ON;
    SET @ErrorLogID = 0;
    BEGIN TRY
        IF ERROR_NUMBER() IS NULL
            RETURN;
        IF XACT_STATE() = -1
        BEGIN
            PRINT 'Cannot log error since the current transaction is in an uncommittable
state. '
                + 'Rollback the transaction before executing uspLogError in order to
successfully log error information.';
            RETURN;
        END
        INSERT [dbo].[ErrorLog]
            (
            [UserName],
            [ErrorNumber],
            [ErrorSeverity],
            [ErrorState],
            [ErrorProcedure],
            [ErrorLine],
            [ErrorMessage]
            )
        VALUES
            (
            CONVERT(sysname, CURRENT_USER),
            ERROR_NUMBER(),
            ERROR_SEVERITY(),
            ERROR_STATE(),
            ERROR_PROCEDURE(),
            ERROR_LINE(),
            ERROR_MESSAGE()
            );
         SET @ErrorLogID = @@IDENTITY;
    END TRY
    BEGIN CATCH
        PRINT 'An error occurred in stored procedure uspLogError: ';
        EXECUTE [dbo].[uspPrintError];
        RETURN -1;
    END CATCH
END;
GO
```

Create Schema [2H]

AUTHORIZATION: Specifies the name of the database-level principal that will own the schema

Create Tables for storing all frequency range experimental data

All tables are stored in the schema [2H] on the "Primary“ filegroup

There are two NOT NULL columns for each table

```
CREATE SCHEMA [2H] AUTHORIZATION [dbo];
GO
CREATE TABLE [2H].[UV](
    [Frequency] [float] NOT NULL,
    [Intensity] [float] NOT NULL
) ON [PRIMARY];
GO

CREATE TABLE [2H].[NIR](
    [Frequency] [float] NOT NULL,
    [Intensity] [float] NOT NULL
) ON [PRIMARY];
GO

CREATE TABLE [2H].[MIR](
    [Frequency] [float] NOT NULL,
    [Intensity] [float] NOT NULL
) ON [PRIMARY];
GO
CREATE TABLE [2H].[FIR_35](
    [Frequency] [float] NOT NULL,
    [Intensity] [float] NOT NULL
) ON [PRIMARY];
GO
```

Load data

Insert values to the table using "BULK INSERT"

```
BULK INSERT [Spectroscopy].[2H].[UV_Al] FROM
N'E:\SQL_Server_Practice\2H\_2H_AL_U.SP_xy.asc'
WITH (
   KEEPIDENTITY,
   TABLOCK
);
```

Alter the table and set a clustered index on Frequency column

Create a non-clustered index to enhance the operating efficiency.

```
ALTER TABLE [2H].[UV_Al] WITH CHECK ADD
    CONSTRAINT [PK_UV_Al_Frequency] PRIMARY KEY
CLUSTERED
    (
        [Frequency]
    )  ON [PRIMARY];
GO

CREATE NONCLUSTERED INDEX [IX_UV_Al_Intensity]
ON [2H].[UV_Al]([Intensity]) ON [PRIMARY];
GO
```

Merge FIR range data

Merge the different FIR frequency range data using “UNION” command

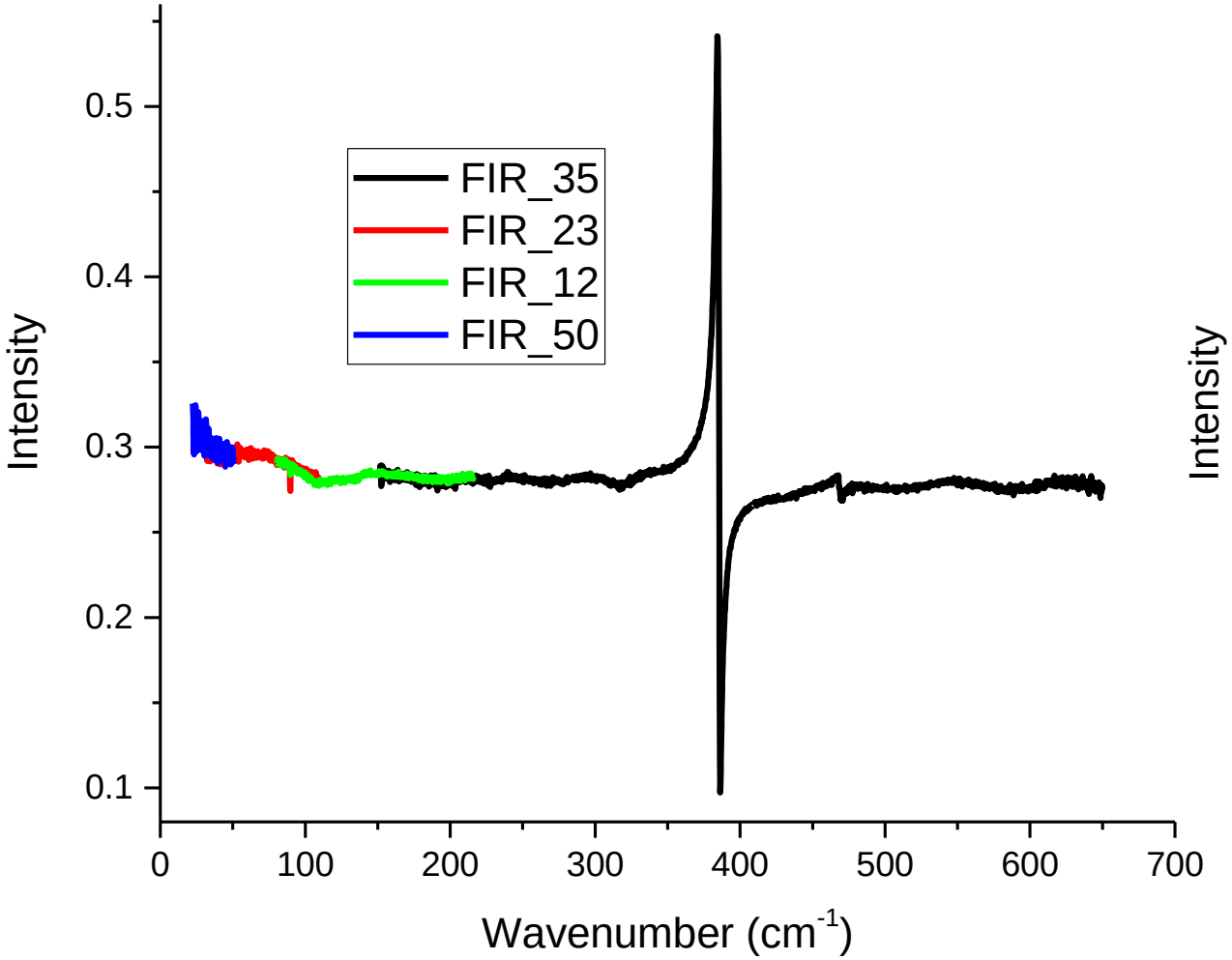
Saved the result to Table “FIR_MERGE”

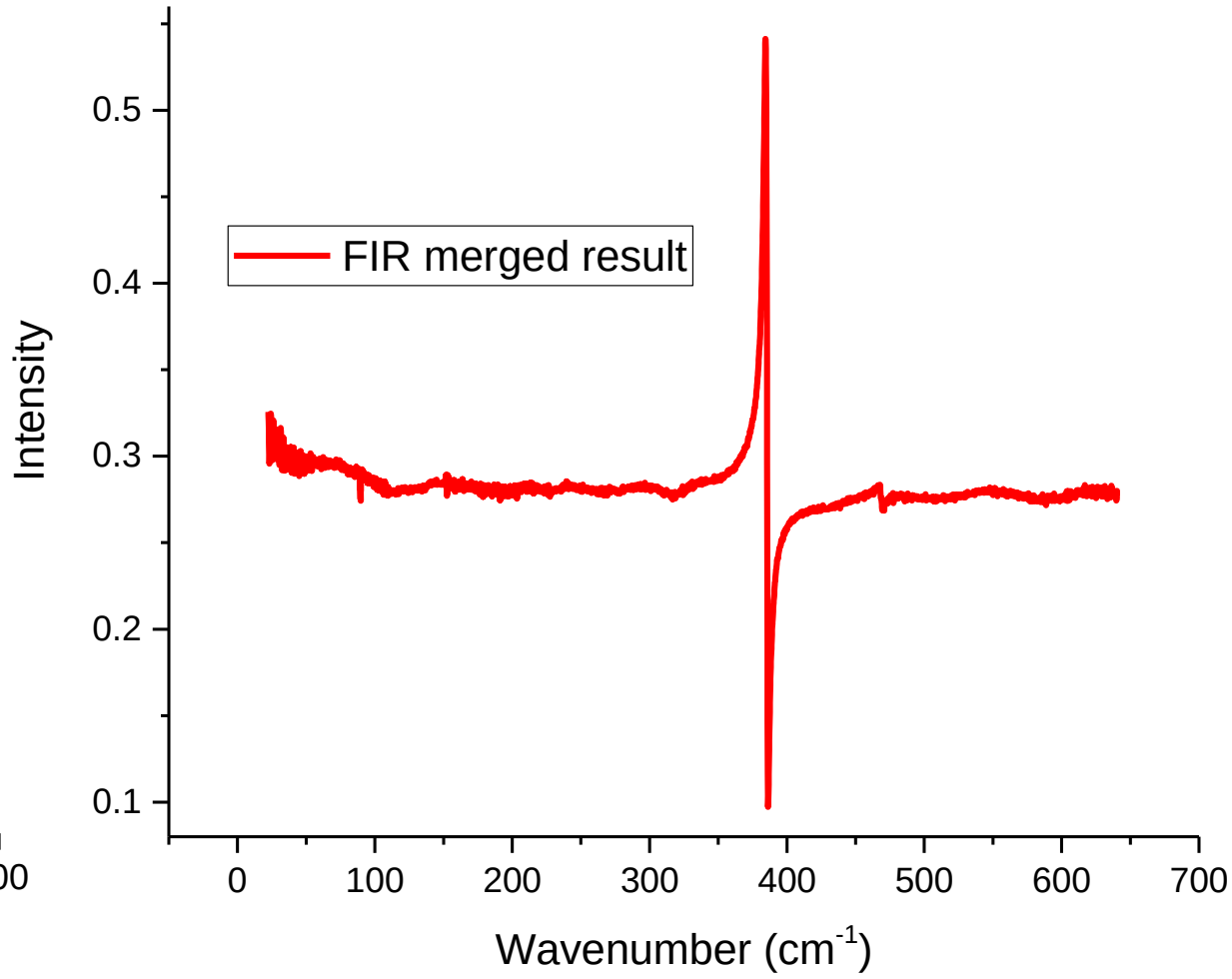
Created a non-clustered index for the new table

```
INSERT INTO [2H].[FIR_MERGE]
SELECT [Frequency], [Intensity] FROM
[2H].[FIR_35]
UNION
SELECT [Frequency], [Intensity] FROM
[2H].[FIR_23]
UNION
SELECT [Frequency], [Intensity] FROM
[2H].[FIR_12]
UNION
SELECT [Frequency], [Intensity] FROM
[2H].[FIR_50];
GO

CREATE NONCLUSTERED INDEX
[IX_FIR_MERGE_Intensity] ON
[2H].[FIR_MERGE]([Intensity]) ON [PRIMARY];
GO
```

FIR merged result





https://github.com/charleseagle/Database-for-merging-and-correcting-data-SQL-Server

Light scattering correction

Conducted the scattering correction using [2H].[UV] divided by [2H].[UV_Al]. The corrected results were saved to a table for the further analysis

Alter the table and set a clustered index on the Frequency column

Created a non-clustered index for the new table

```
INSERT INTO [Spectroscopy].[2H].[UV_Corrected]
SELECT [2H].[UV].Frequency,
[2H].[UV].Intensity/[2H].[UV_Al].Intensity AS
Intensity
FROM [2H].[UV], [2H].[UV_Al]
WHERE [2H].[UV].Frequency =
[2H].[UV_Al].Frequency;

ALTER TABLE [2H].[UV_Corrected] WITH CHECK ADD
    CONSTRAINT [PK_UV_Corrected_Frequency]
PRIMARY KEY CLUSTERED
    (
        [Frequency]
    )  ON [PRIMARY];
GO

CREATE NONCLUSTERED INDEX
[IX_UV_Corrected_Intensity] ON
[2H].[UV_Corrected]([Intensity]) ON [PRIMARY];
GO
```

Merge all frequency range data

Calculated the spectrum shift using FIR data as the baseline and the result assign to the parameters

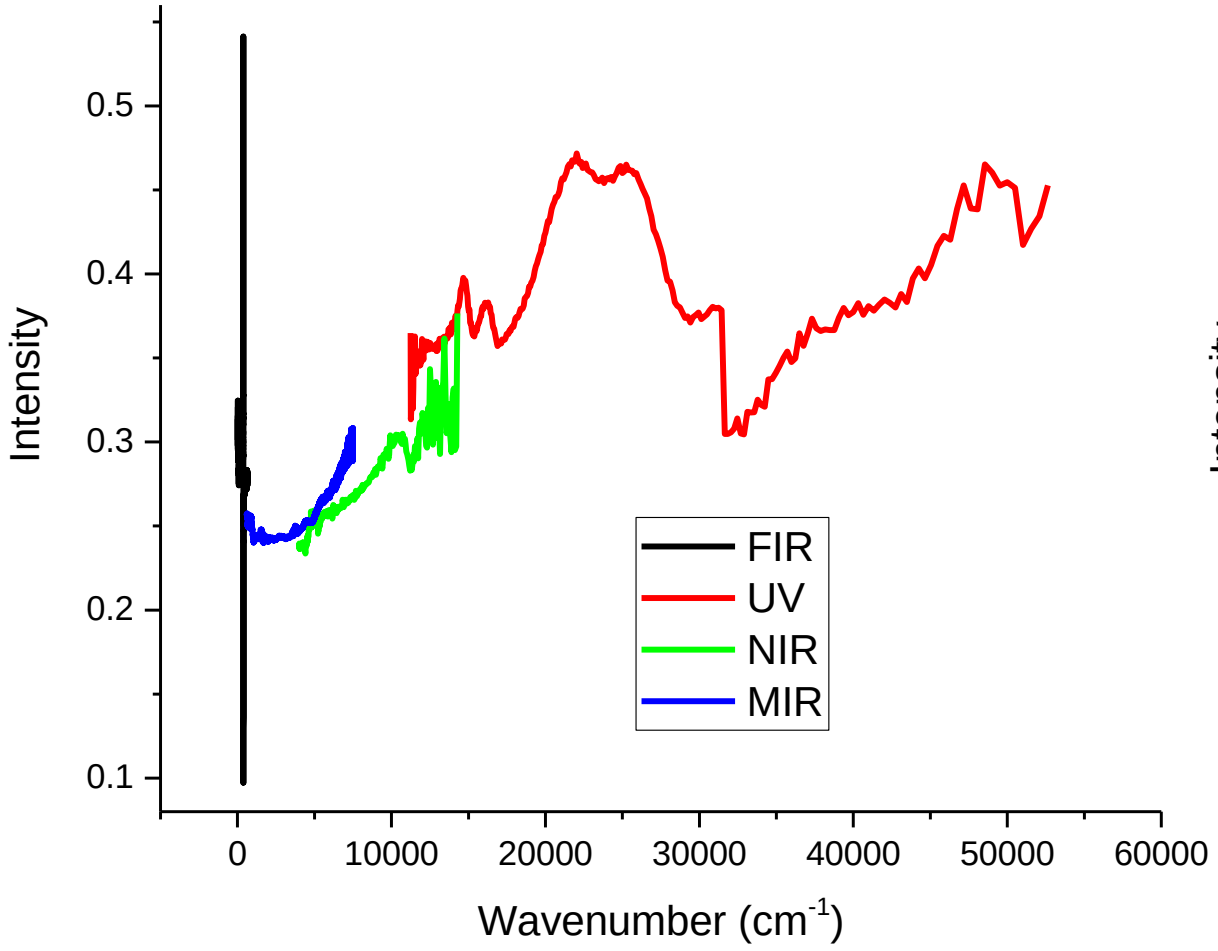
Merge the different frequency range data using “UNION ALL” command

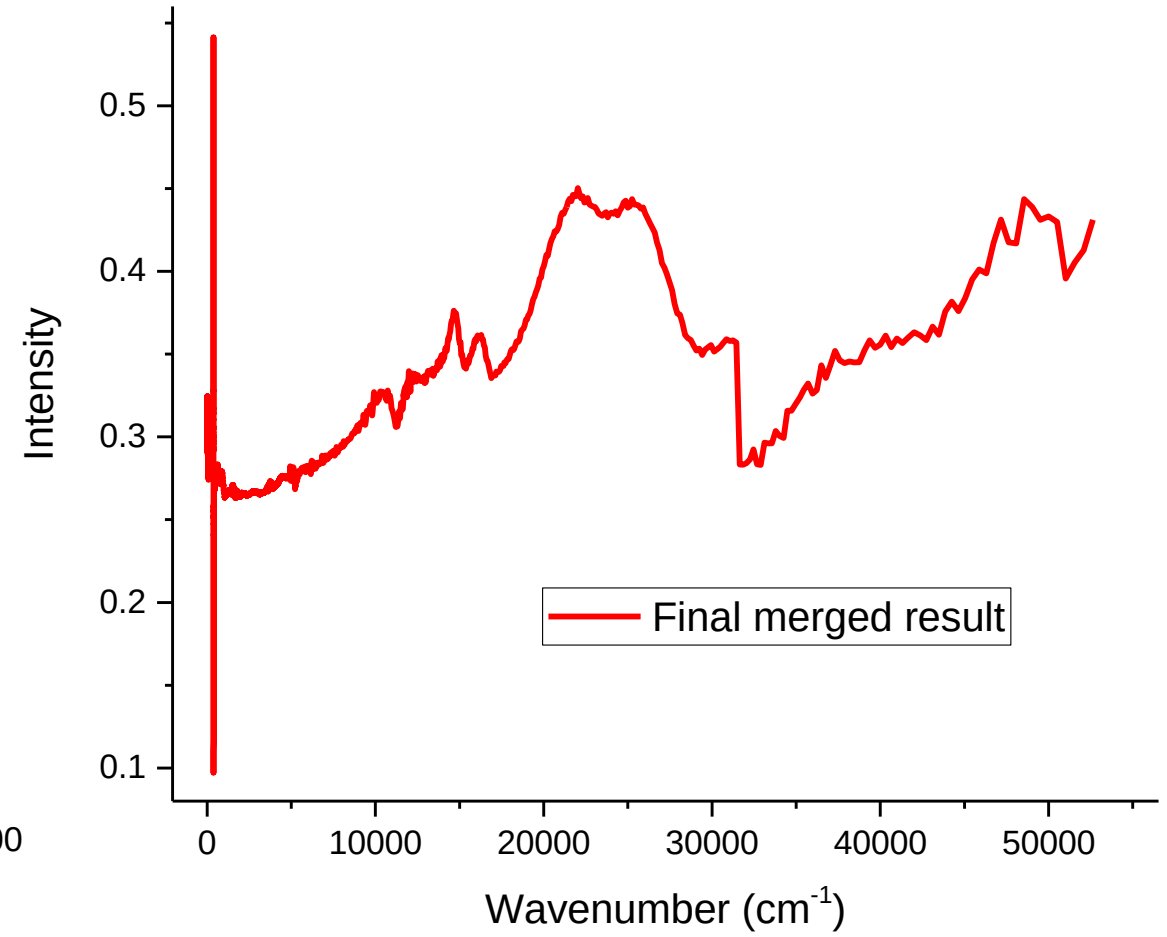
Saved the results to Table “Merge”

```
DECLARE @MIR float = (SELECT A.[Intensity] - B.[Intensity] AS Diff
FROM [2H].[FIR_MERGE] AS A, [2H].[MIR_Corrected] AS B
WHERE A.[Frequency] = 640.37 AND B.[Frequency] = 640.3677)
, @NIR float = (SELECT A.[Intensity] - B.[Intensity] AS Diff
FROM [2H].[MIR_Corrected] AS A, [2H].[NIR_Corrected] AS B
WHERE A.[Frequency] = 4930.05973 AND B.[Frequency] = 4930.966469)
, @UV float = (SELECT A.[Intensity] - B.[Intensity] AS Diff
FROM [2H].[NIR_Corrected] AS A, [2H].[UV_Corrected] AS B
WHERE A.[Frequency] = 11682.24299 AND B.[Frequency] = 11655.01166)
;

INSERT INTO [2H].[MERGE]
SELECT [Frequency], [Intensity] FROM [2H].[FIR_MERGE]
UNION ALL
SELECT [Frequency], [Intensity] + @MIR AS Intensity
FROM [2H].[MIR_Corrected]
UNION ALL
SELECT [Frequency], [Intensity] + @MIR + @NIR AS Intensity
FROM [2H].[NIR_Corrected]
UNION ALL
SELECT [Frequency], [Intensity] + @MIR + @NIR + @UV AS Intensity
FROM [2H].[UV_Corrected];
GO
```

Final merged result





https://github.com/charleseagle/Database-for-merging-and-correcting-data-SQL-Server

Website demos: flexible design

Qi Sun (Charles)

Home | Resume | Certificates | GitHub and Linkedin | Experience Summary | Publications

NEWS

My experience summary.

12/29/2015 - I created an online shopping

INTRODUCE MYSELF

I was born in Huadian city, Jilin provice, China. A beautiful city lies in the northeast of China.

I got my PhD degree in Physical Chemistry from University of Tennessee in December 2012.

MY EXPERIENCE

I have a real passion for database technology and programming.

I spent most of my time to analyze the data, and found science behind them using Python and SQL Server. Because I measured the samples using different instruments which cover different frequency ranges, I need to merge the data. I also need to correct data from the scattering and convert data into different units. SQL Server can quickly finish these assignments and helped me save a lot of time on data analysis. From these processes, I practiced a lot of SQL commands, database building, maintenance, and backup. I also analyzed data using different numerical models such as Lorentz model, Gaussian, Kramers-Kronig model. I chose Python as the programming language because it is free and has good libraries. The data presentation was done by many tools such as Origin, Excel, Photoshop, and Diamond. Beautiful figures are always required by high quality journals. During my PhD study, I got well trained on data analysis and presentation.

After my PhD study at University of Tennessee, I joined University of Colorado, Boulder as a Research Associate. I worked on very different projects. At work I use Matlab as data analysis tool because it is very

Website demos: flexible design

Home | Resume | Certificates | GitHub and Linkedin | Experience Summary | Publications

CERTIFICATES

Microsoft® Certified Solutions Expert: Business Intelligence

Microsoft® Certified Solutions Associate: SQL Server 2012

Microsoft Certified Solutions Developer: Web Applications

Microsoft Specialist: Programming in HTML5 with JavaScript and CSS3

Microsoft Specialist: Programming in C#

Microsoft Technology Associate: Database Fundamentals

Microsoft Certified Professional

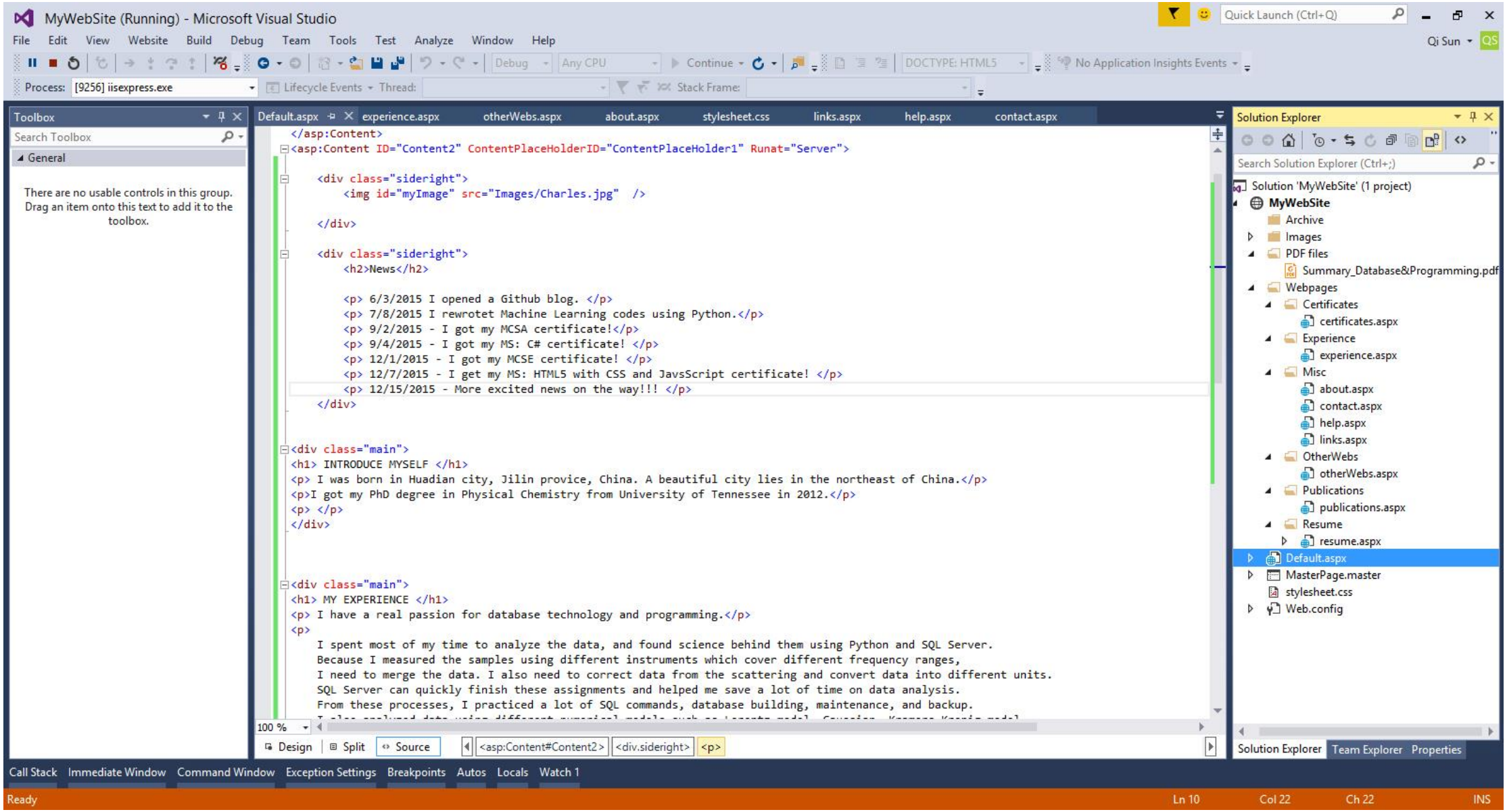
Qi Sun (Charles)

Home | Resume | Certificates | GitHub and Linkedin | Experience Summary | Publications

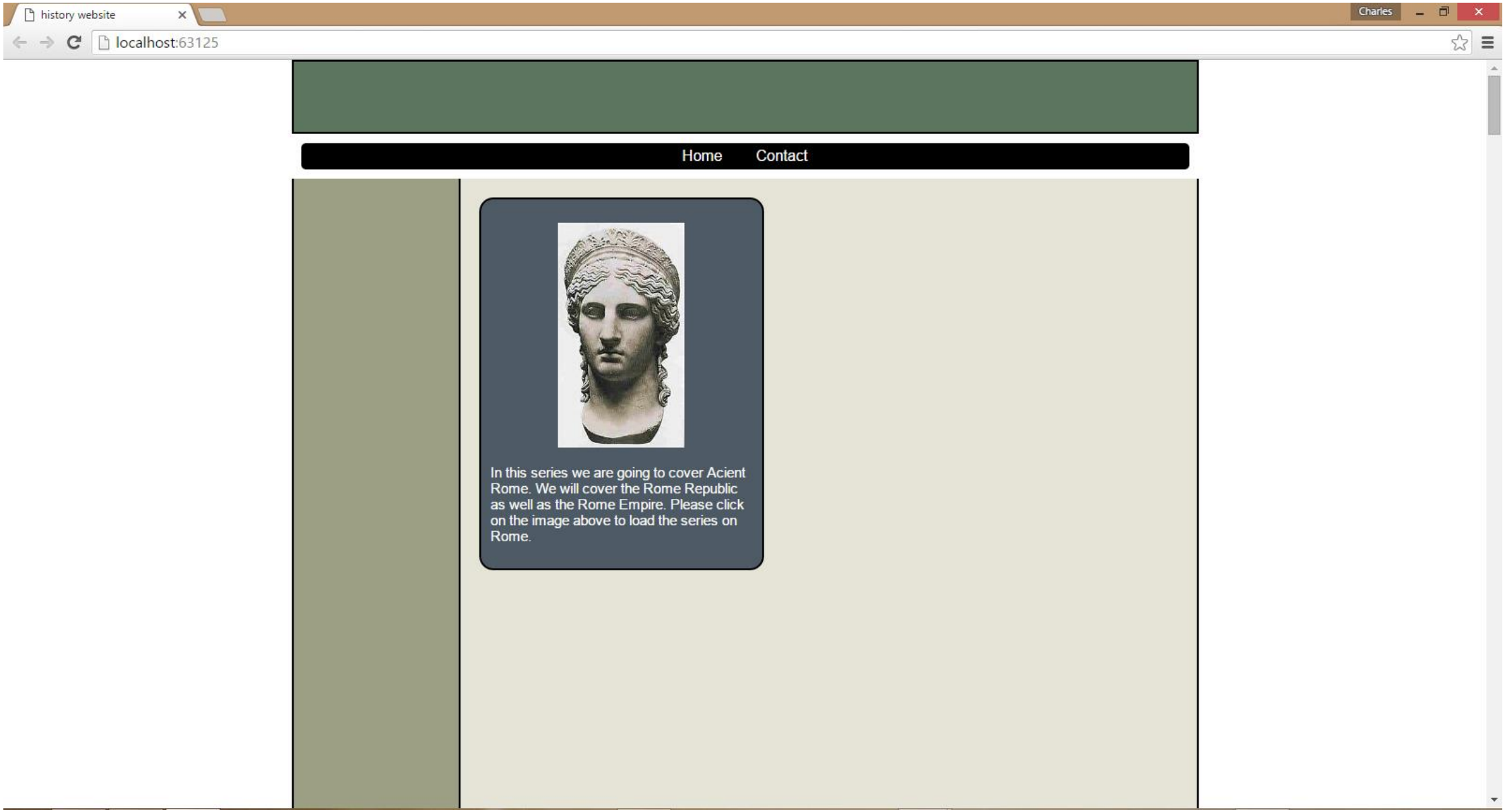
- Summary_Database&Programming.pdf
- Hotel booking dystem with database
- Online shopping system with database.
- DVD rental database using C# with two windows forms as inputs.
- Machine Learning Python codes.

About | Useful Links | Help | Contact

Website demos: ASP.NET code demos

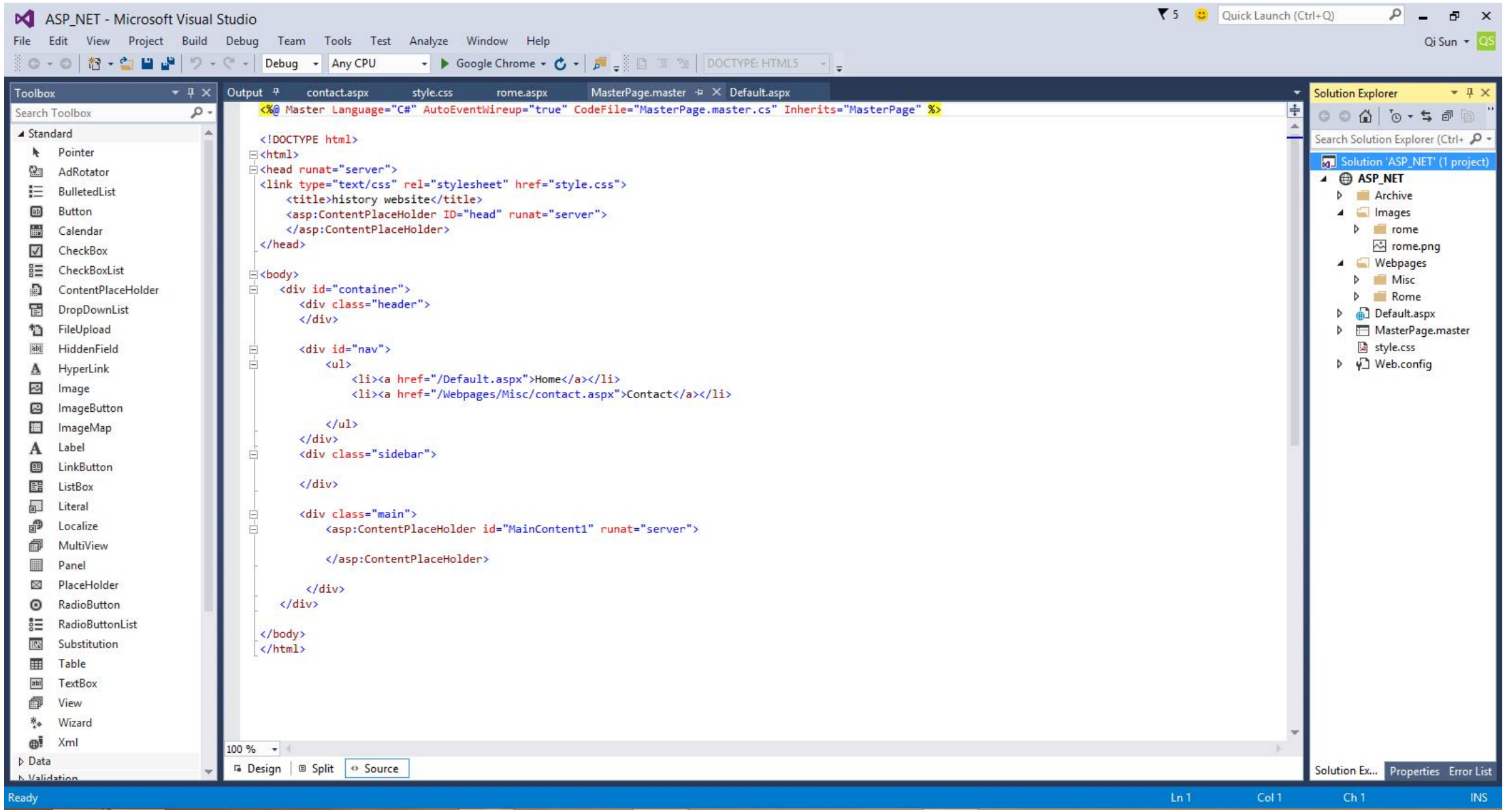


Website demos: fixed design



https://github.com/charleseagle/ASP.NET-Websites

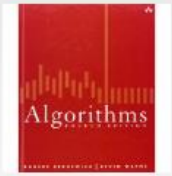
Website demos: ASP.NET code demos



https://github.com/charleseagle/ASP.NET-Websites

Website demos: online shopping

Good Shop | Category | Search | Sign in | Your cart: 0 item(s), $0.00 | Cart

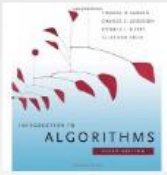


Algorithms

Good book Weight: 3 pounds

Quantity: 8 $63.08

Add to Cart

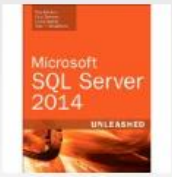


Introduction to Algorithms

Good book Weight: 3 pounds

Quantity: 5 $58.08

Add to Cart



Microsoft SQL Server 2014 Unleashed

Good book Weight: 2 pounds

Quantity: 12 $43.88

Add to Cart

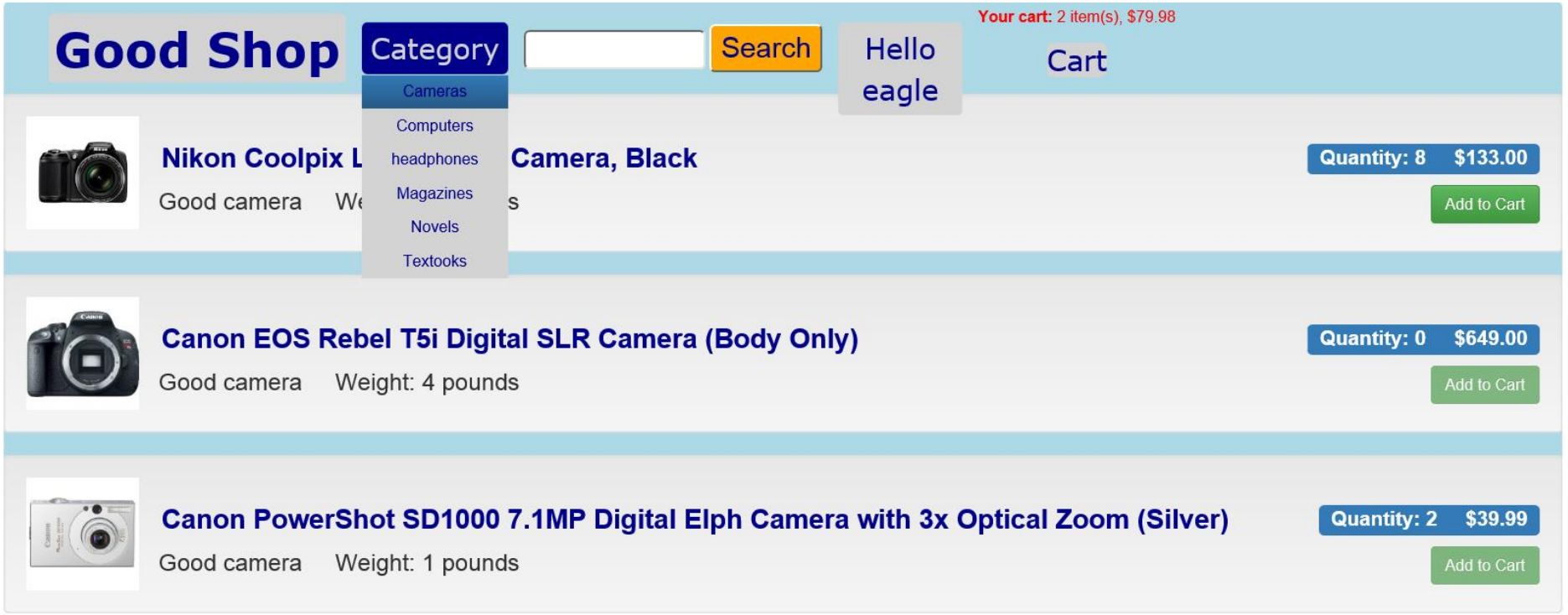
Training Kit (Exam 70-461) Querying Microsoft SQL Server 2012

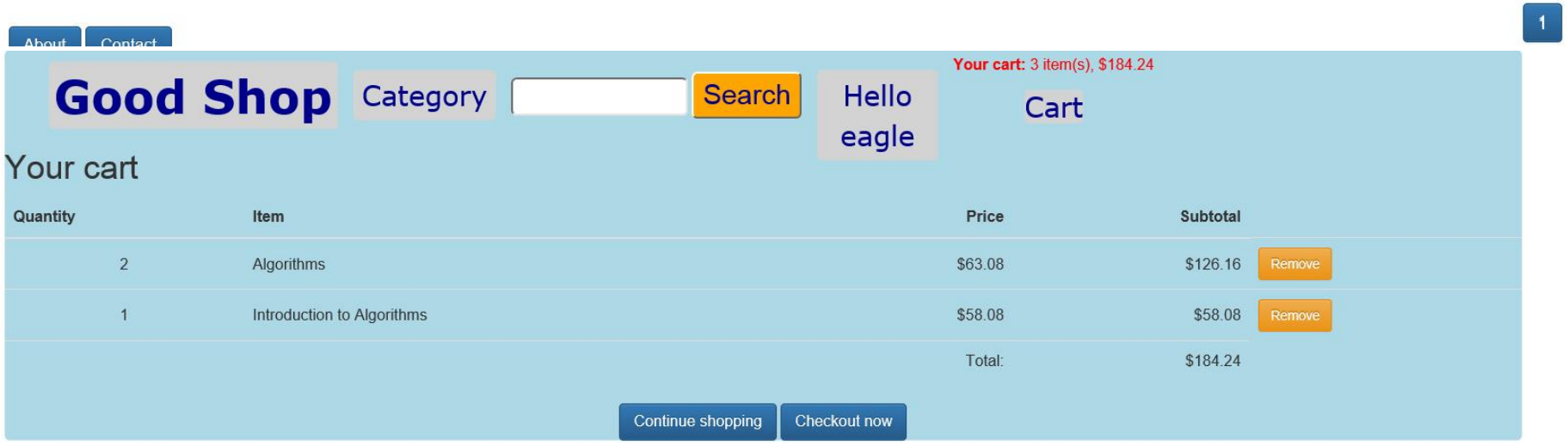
Good book Weight: 3 pounds

Quantity: 0 $44.34

Add to Cart

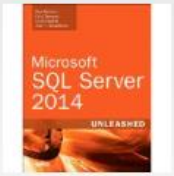
Website demos: online shopping





Website demos: online shopping

Good Shop | Category | SQL | Search | Sign in | Your cart: 0 item(s), $0.00 | Cart



Microsoft SQL Server 2014 Unleashed

Good book Wieght: 2 pounds

Quanityt: 11 $43.88

Add to Cart

Training Kit (Exam 70-461) Querying Microsoft SQL Server 2012

Good book Wieght: 3 pounds

Quanityt: 3 $44.34

Add to Cart

Training Kit (Exam 70-462) Administering Microsoft SQL Server 2012 Databases

Good book Wieght: 3 pounds

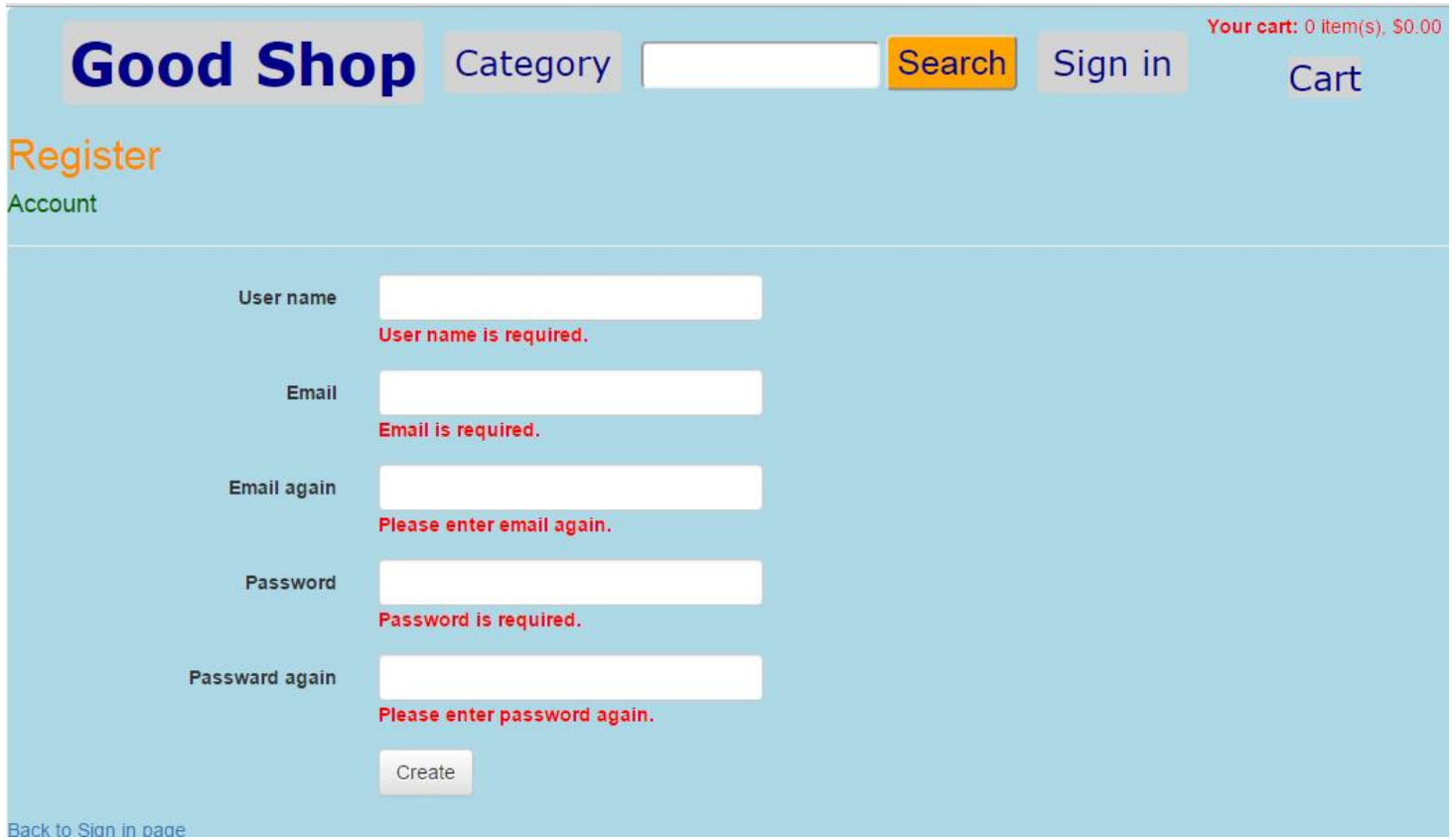
Quanityt: 3 $47.28

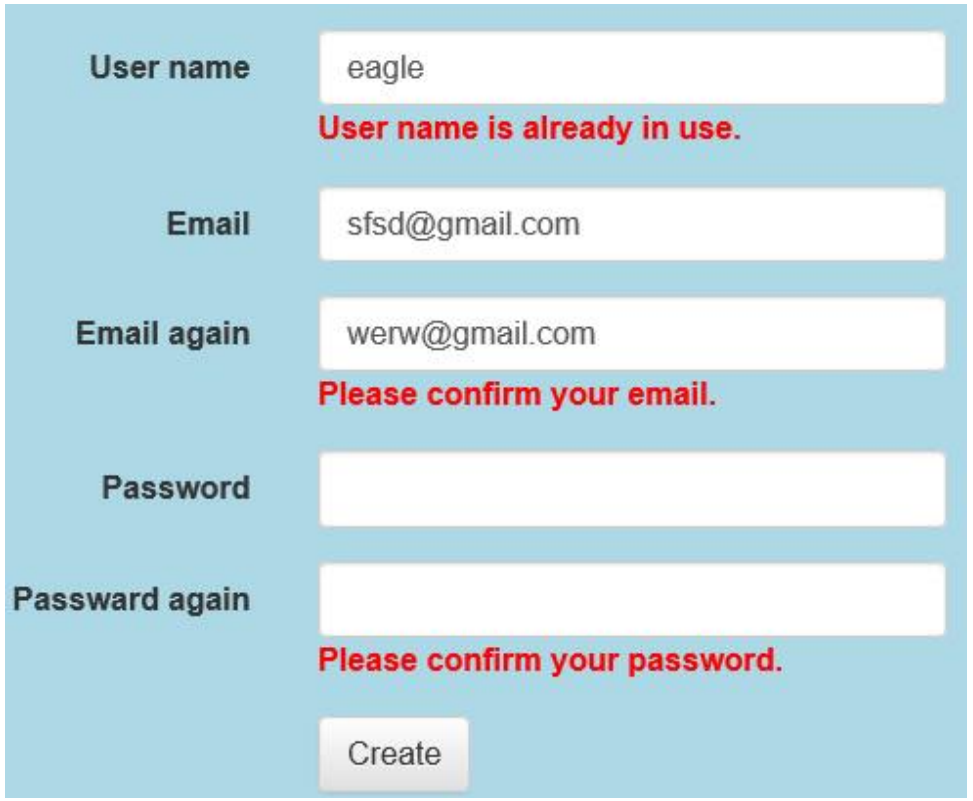
Add to Cart

1

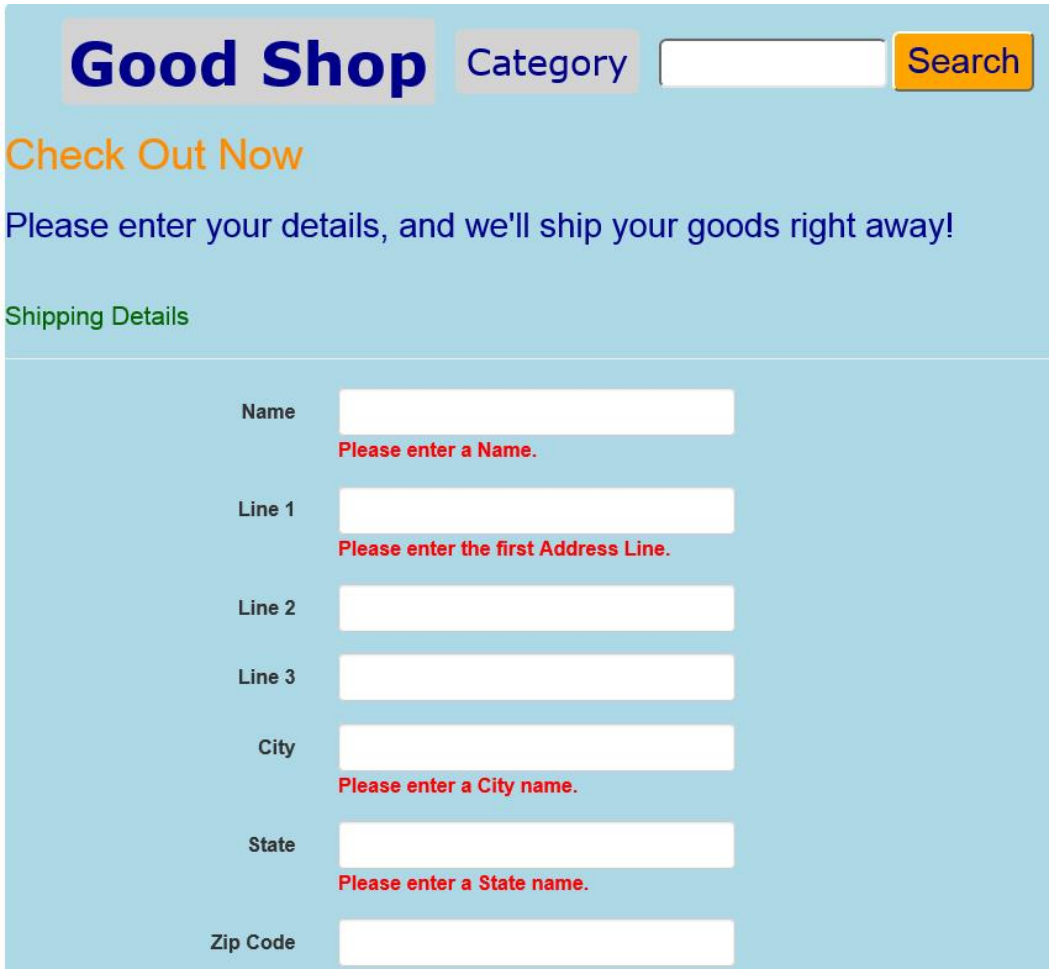
About | Contact

Website demos: online shopping





Website demos: online shopping



Website demos: online shopping

Good Shop

Create

Product

Name	
Quantity	0
Price	0.00
Color	
ProductWeight	0
ProductWeightUnit	
Category	
Description	

Create

Good Shop

charleseagleLogOut

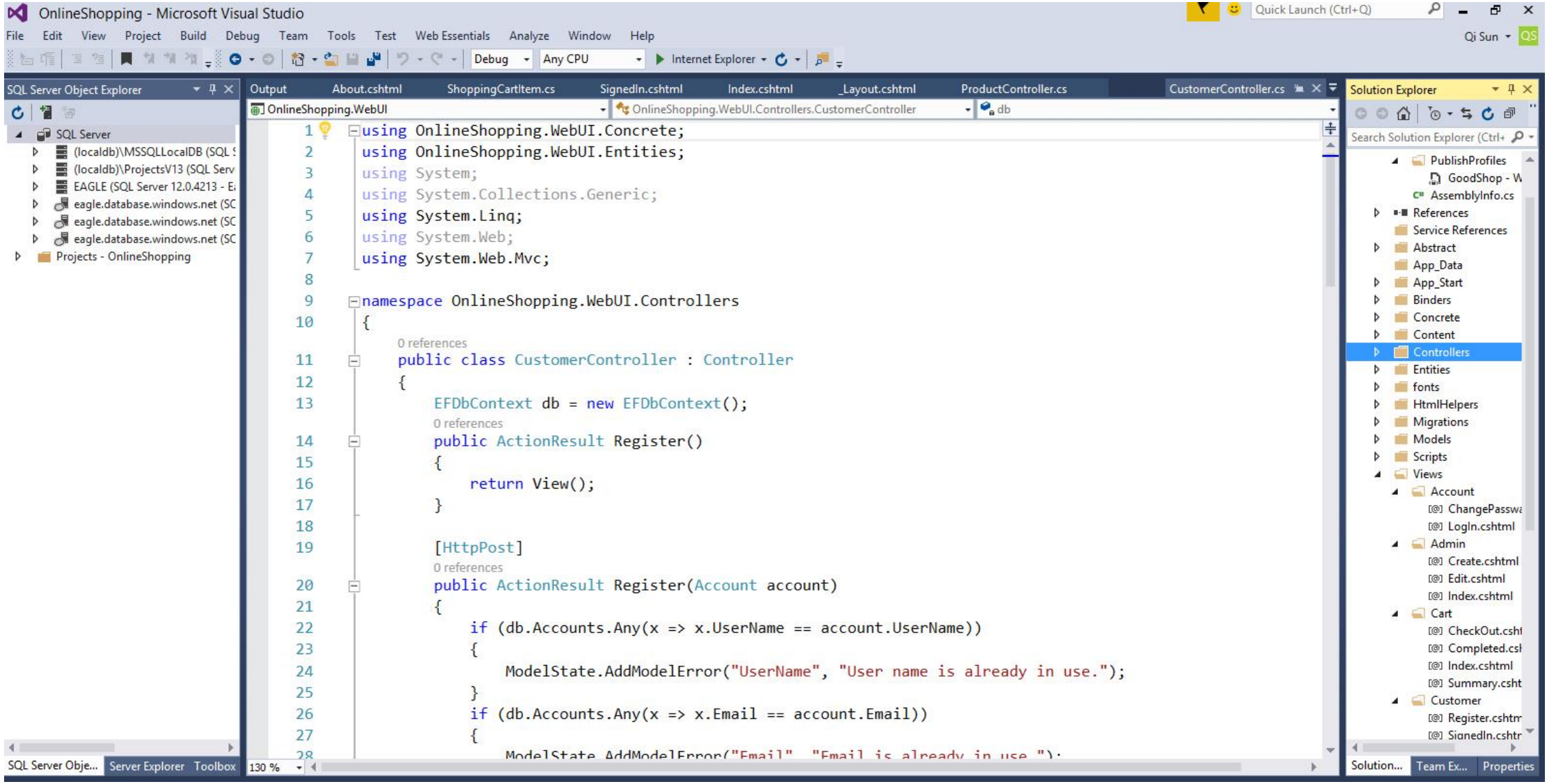
All Products

ID	Name	Price	Actions
1	Algorithms	$63.08	Delete
2	Introduction to Algorithms	$58.08	Delete
3	Microsoft SQL Server 2014 Unleashed	$43.88	Delete
4	Training Kit (Exam 70-461) Querying Microsoft SQL Server 2012	$44.34	Delete
5	Training Kit (Exam 70-462) Administering Microsoft SQL Server 2012 Databases	$47.28	Delete
6	Training Kit (Exam 70-463) Implementing a Data Warehouse	$45.08	Delete
7	Expert ASP.NET Web API 2 for MVC Developers	$43.08	Delete
8	ASP.NET MVC 5 with Bootstrap and Knockout.js	$23.08	Delete

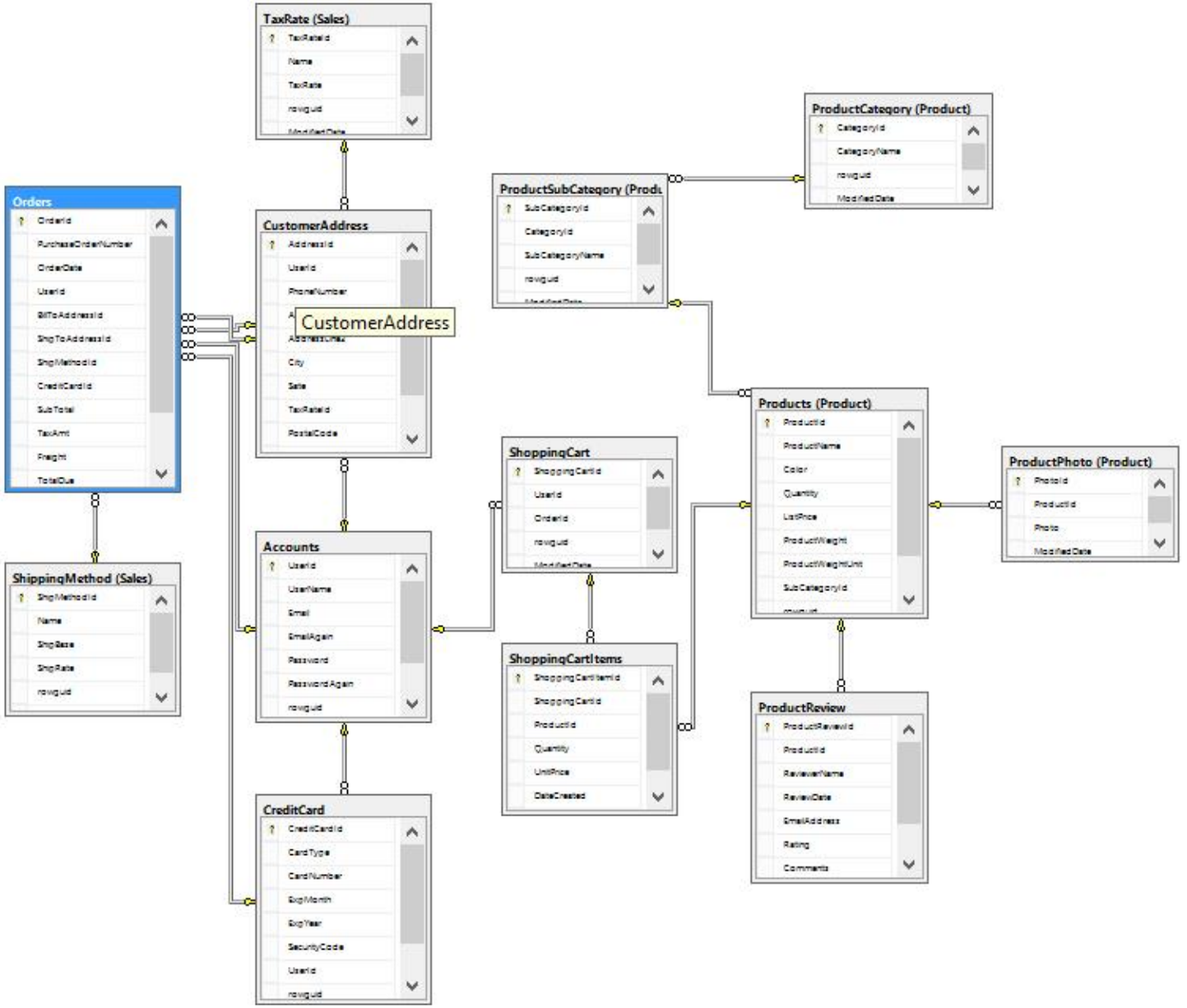
37	JBL J33a BLK Premium In Ear Headphones with JBL Drivers and Microphone, Black	$22.05	Delete
38	Yurbuds (CE) Inspire 300 Noise Isolating In-Ear Headphones, Mossy Oak Orange	$23.00	Delete
39	Panasonic RP-HJE125E-V Headphone	$10.00	Delete

Add a New Product

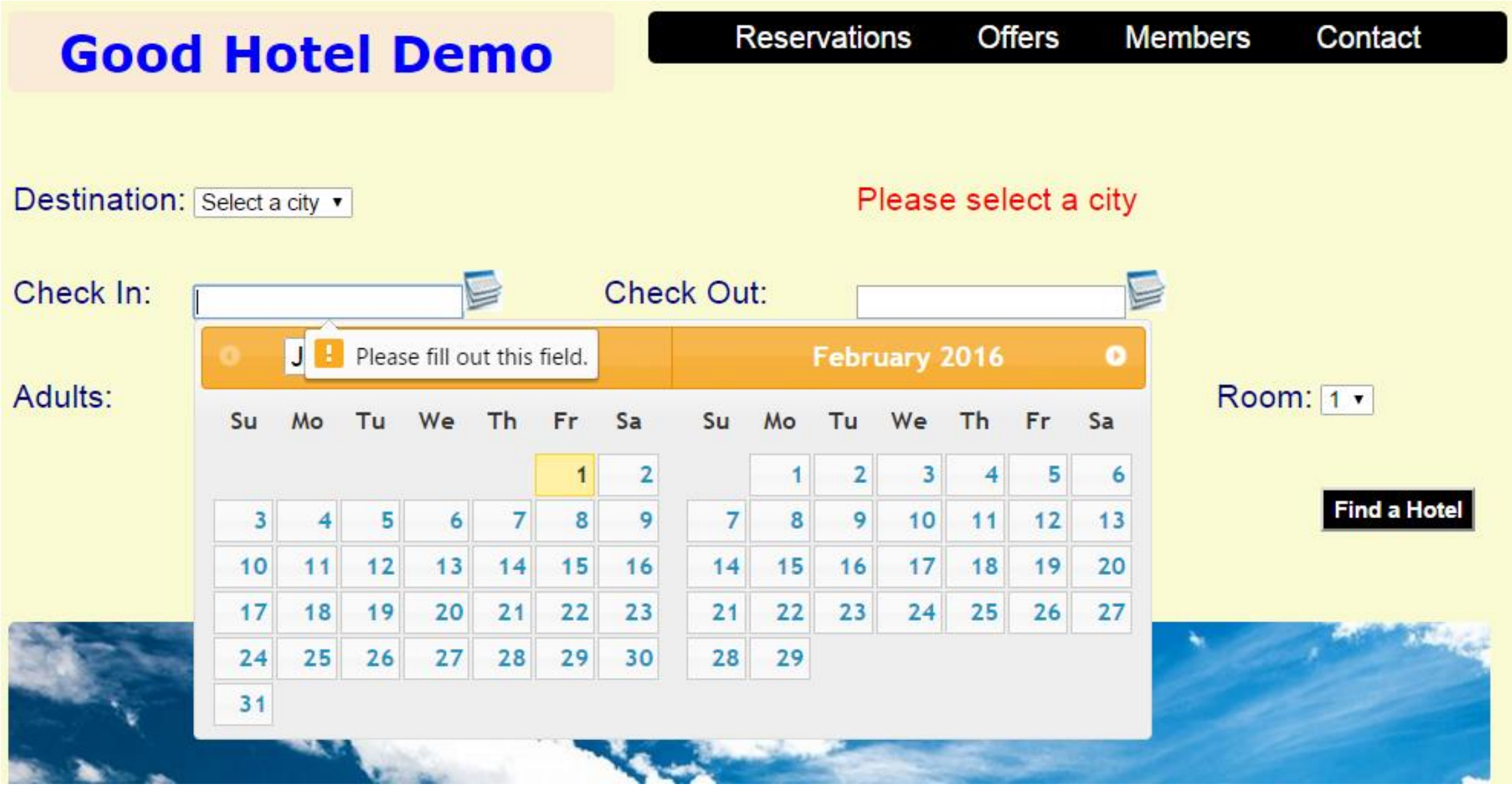
Website demos: online shopping



Website demos: online shopping



Website demos: online hotel booking



Website demos: online hotel booking

Good Hotel Demo

Reservations Offers Members Contact

STANDARD ROOM

Availability: 3 rooms left!

Price: $ 80

Member Price: $ 72

Description: The hotel has 521 spacious guestrooms & suites featuring free Wi-Fi.

Book this room

1 QUEEN BED

Availability: 1 rooms left!

Price: $ 100

Member Price: $ 90

Description: The hotel has 521 spacious guestrooms & suites featuring free Wi-Fi.

Book this room

2 KING BED

Availability: 1 rooms left!

Price: $ 200

Member Price: $ 180

Description: The hotel has 521 spacious guestrooms & suites featuring free Wi-Fi.

Book this room

Website demos: online hotel booking

Website demos: online hotel booking

Good Hotel Demo

Reservations | Offers | Members | Contact

Congratulations Qi Sun !

Your order processd sucessfully.

Enjoy your upcoming trip!

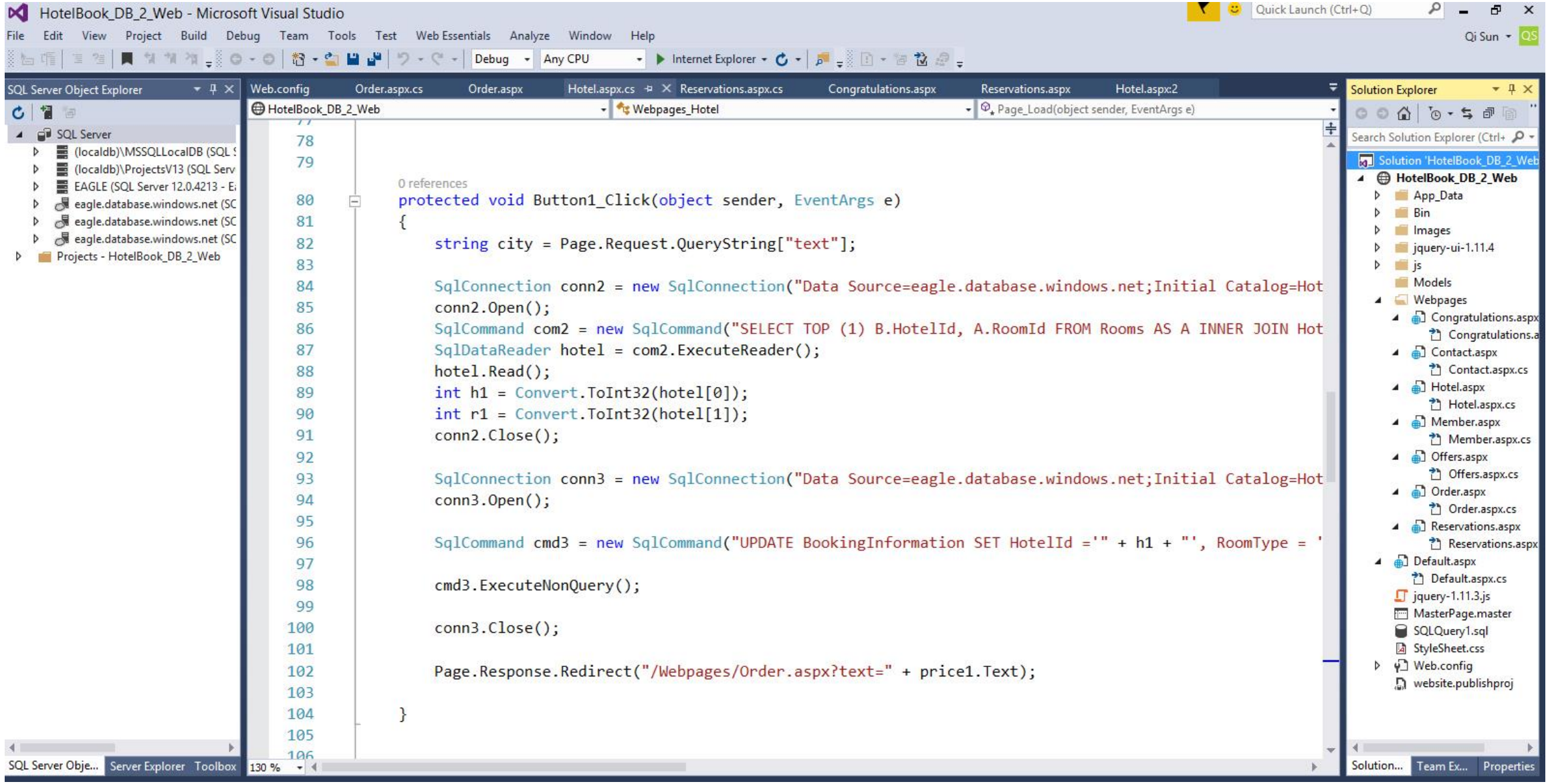
PLease remember our order number: 25 !!!

Hotel Name:	Mart Plaza River North
Room Number:	1001
Check In:	12/30/2015 12:00:00 AM
Check Out	12/31/2015 12:00:00 AM
Final Price:	$ 100

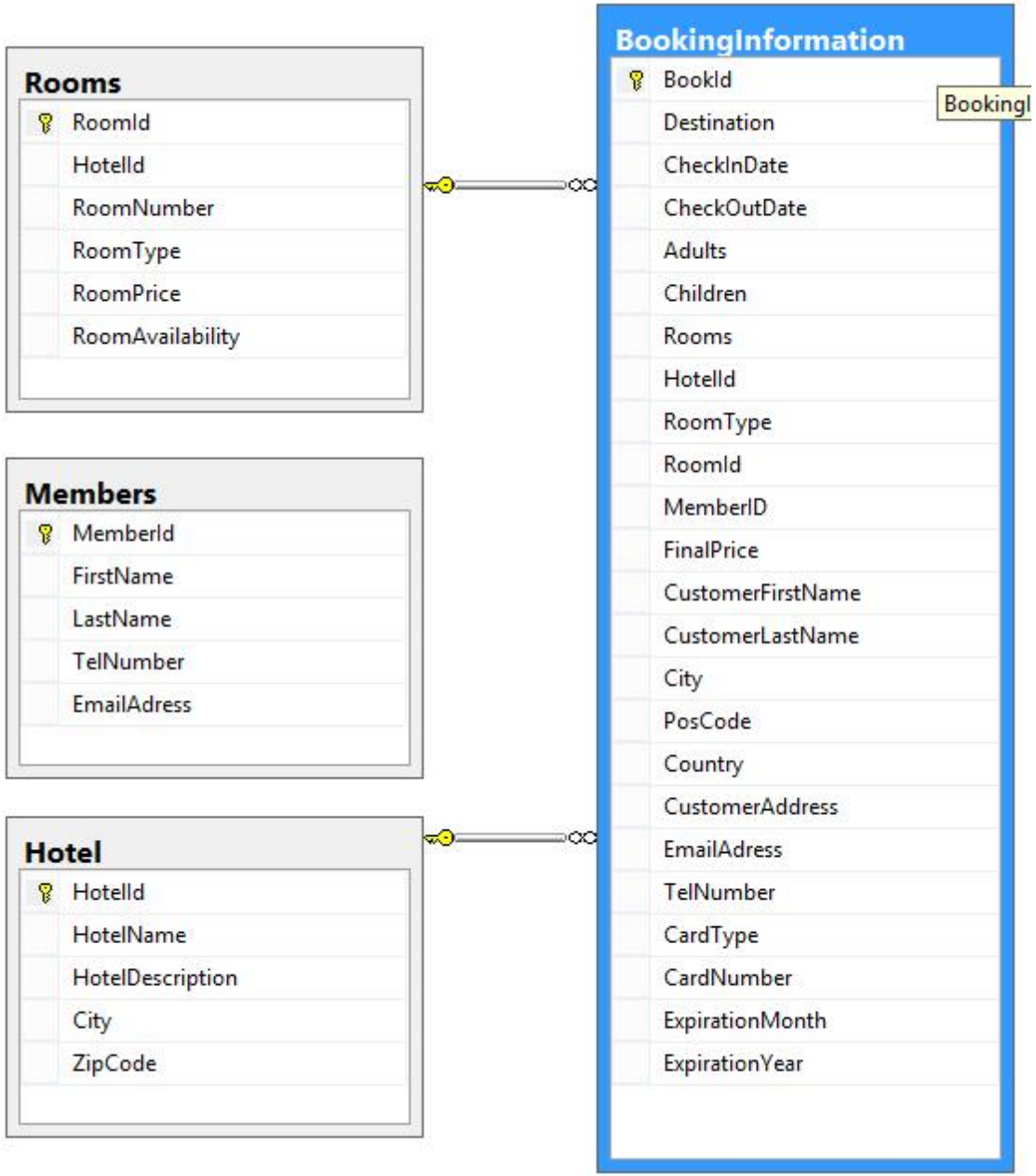
We also sent you a confirmation email to sunqicharles@gmail.com
Please check your emamil box.

Website demos: online hotel booking

Website demos: online hotel booking

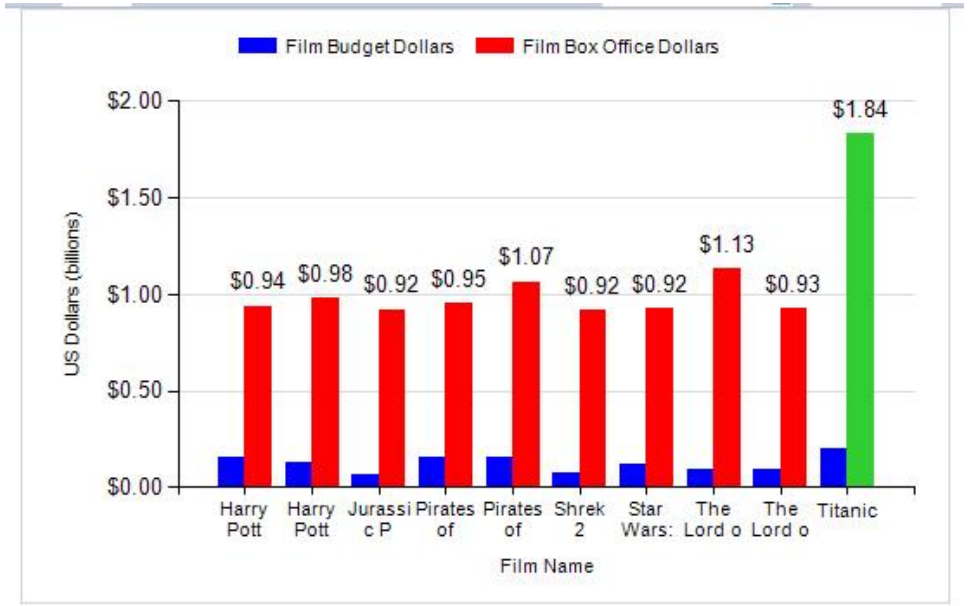


Website demos: online hotel booking



SSRS demos

Chart

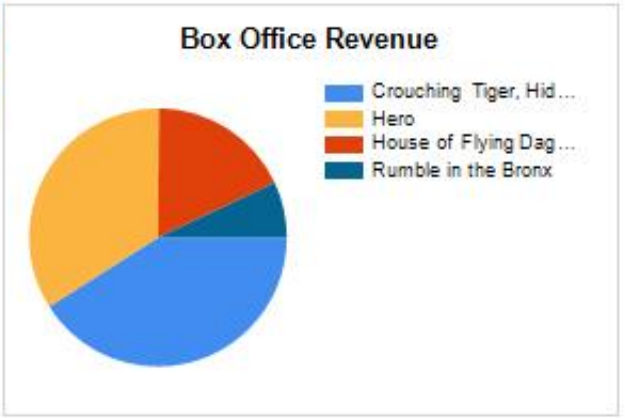


Gauge

Film Name	Nomination	Wins	Indicator
Jurassic Park	3	3	
Spider-Man	2	0	
King Kong	4	3	
Superman Returns	1	0	

List

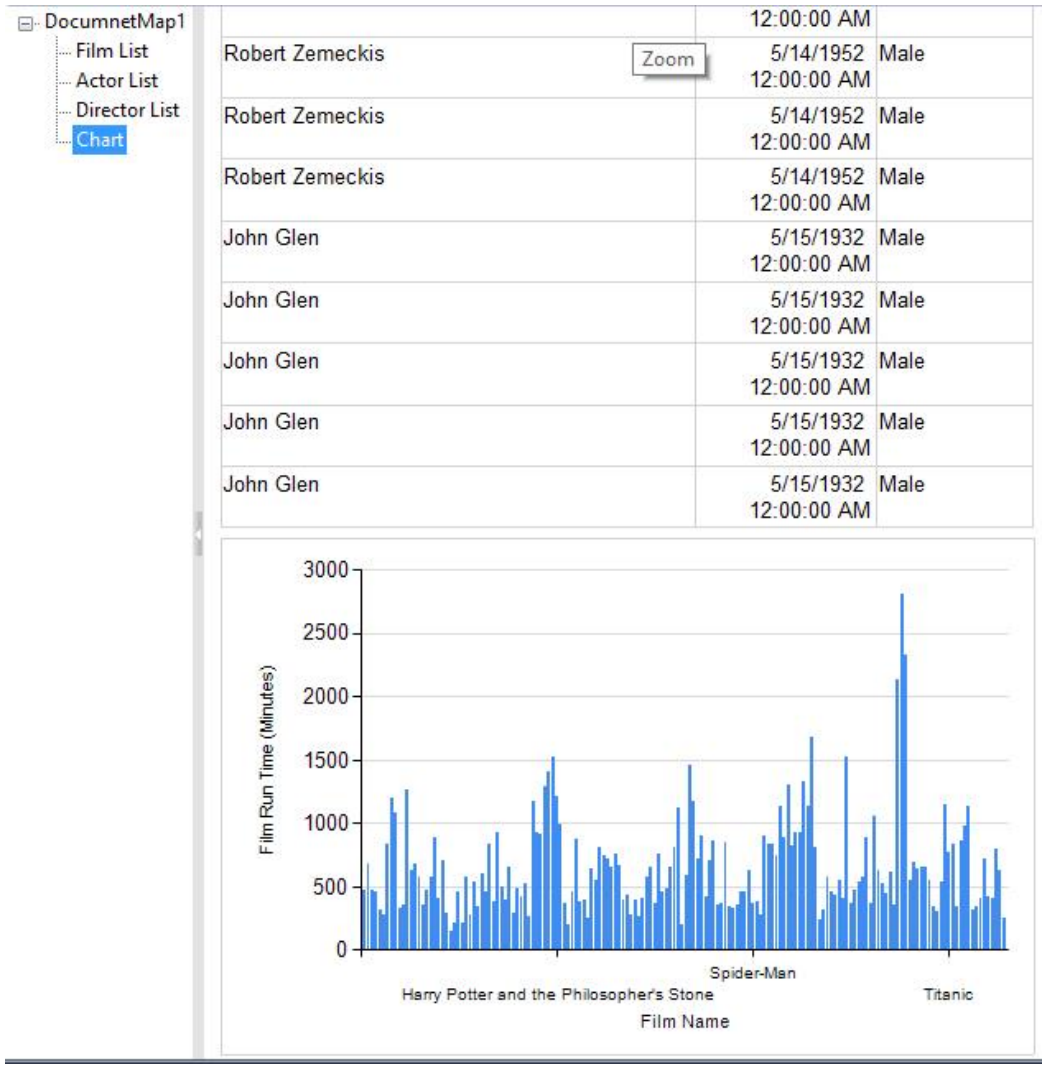
China	
Film Name	**Film Release Date**
Hero	09/24/2004
Crouching Tiger, Hidden Dragon	01/05/2001
House of Flying Daggers	01/14/2005
Rumble in the Bronx	07/04/1997



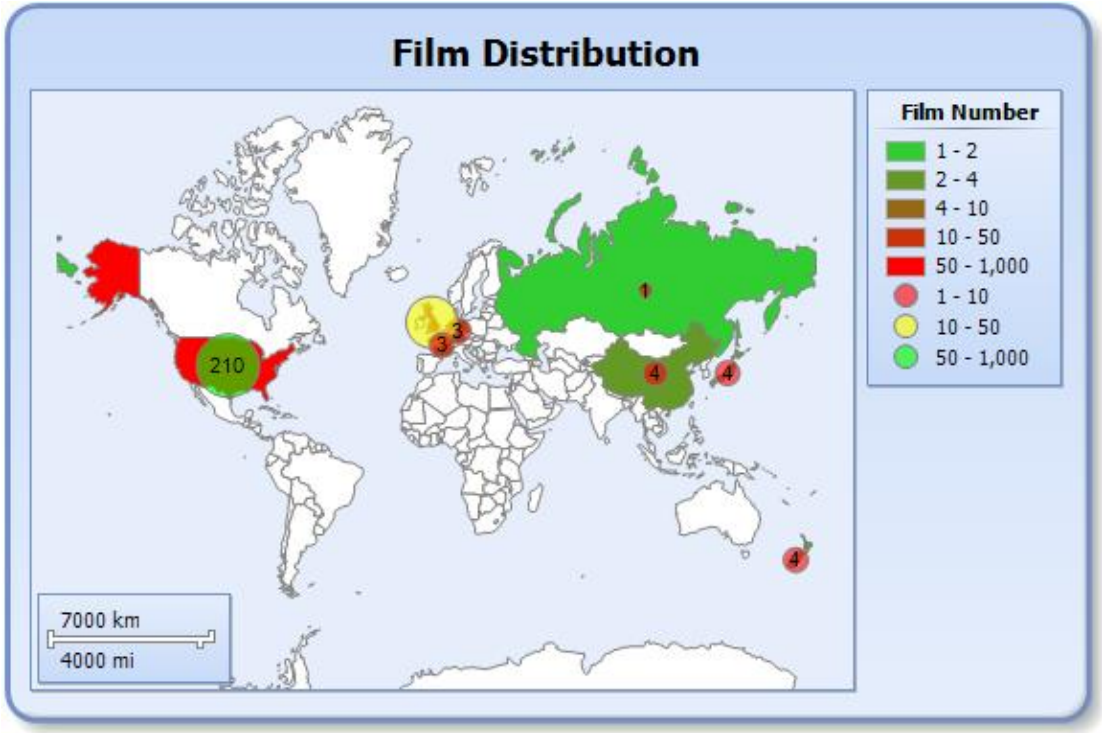
https://github.com/charleseagle/SQL-Server-SSRS

SSRS demos

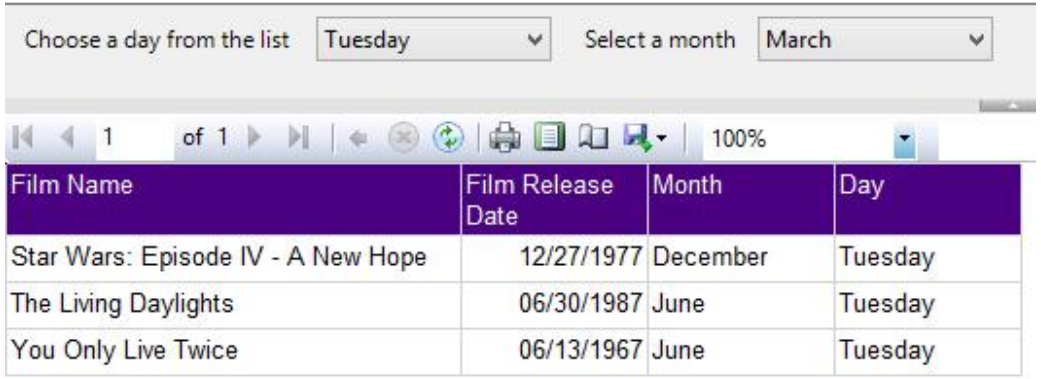
Document Map



Map



Dropdown Parameters



Film Name	Film Release Date	Month	Day
Star Wars: Episode IV - A New Hope	12/27/1977	December	Tuesday
The Living Daylights	06/30/1987	June	Tuesday
You Only Live Twice	06/13/1967	June	Tuesday

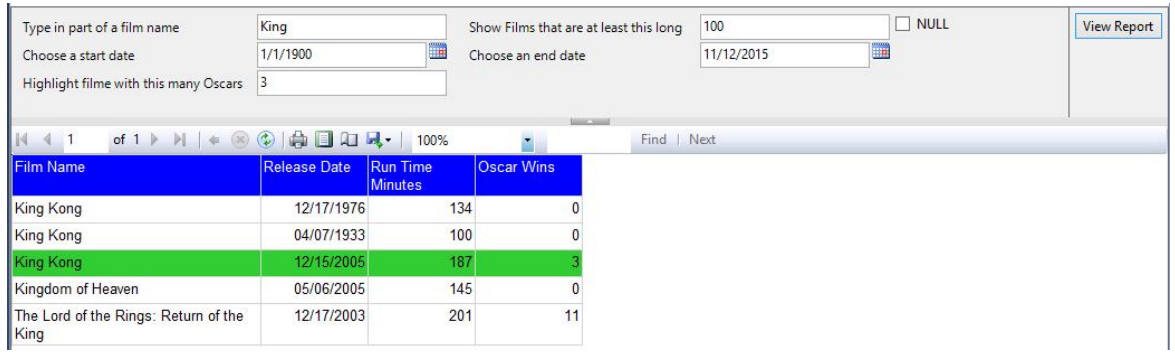
https://github.com/charleseagle/SQL-Server-SSRS

SSRS demos

Matrix

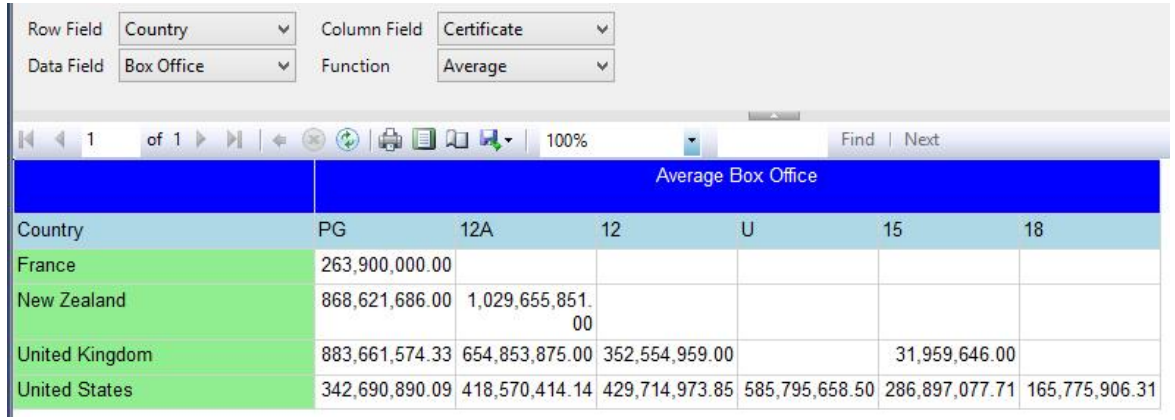
	Film certificate								
Studio Name	12		12A		15		18		PG
	Total	Average	Total	Average	Total	Average	Total	Average	Total
Warner Bros. Pictures	376	125.33	988	141.14	1430	130.00	151	151.00	
Walt Disney Pictures	143	143.00	318	159.00					
Universal Pictures	271	135.50	774	129.00	355	118.33	335	167.50	
Touchstone Pictures	549	137.25	478	119.50	132	132.00	139	139.00	
Revolution Studios			122	122.00	124	124.00			
Paramount Pictures	485	121.25	471	117.75	522	130.50	96	96.00	
New Line Cinema	90	90.00	471	157.00	119	119.00	357	119.00	
Morgan Creek Productions	112	112.00							
Miramax Films							414	138.00	
MGM	377	125.67	277	138.50	134	134.00			
Lucasfilm			140	140.00					
Jerry Bruckheimer Films					396	132.00			
Imagine Entertainment			100	100.00					
Gaumont									
Dreamworks	288	144.00	144	144.00	738	147.60			
Disney Pixar									
Columbia Pictures			872	124.57	287	143.50			
Carolco Pictures					250	125.00	113	113.00	
Avalon Studios							99	99.00	
20th Century Fox	443	147.67	510	127.50	1036	129.50	399	133.00	
Total	3134	130.58	5665	131.74	5523	131.50	2103	131.44	

Pivot table



Film Name	Release Date	Run Time Minutes	Oscar Wins
King Kong	12/17/1976	134	0
King Kong	04/07/1933	100	0
King Kong	12/15/2005	187	3
Kingdom of Heaven	05/06/2005	145	0
The Lord of the Rings: Return of the King	12/17/2003	201	11

Parameters

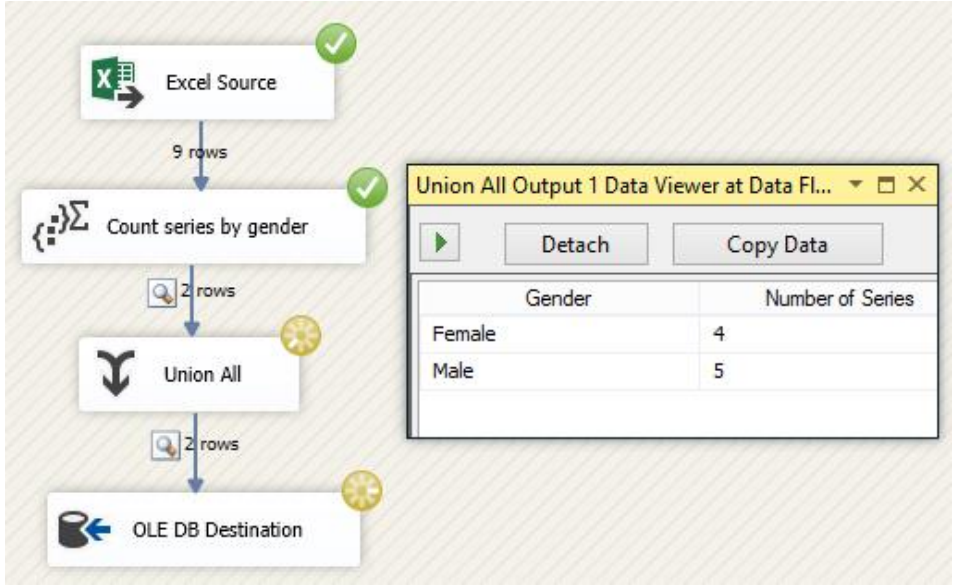


	Average Box Office					
Country	PG	12A	12	U	15	18
France	263,900,000.00					
New Zealand	868,621,686.00	1,029,655,851.00				
United Kingdom	883,661,574.33	654,853,875.00	352,554,959.00		31,959,646.00	
United States	342,690,890.09	418,570,414.14	429,714,973.85	585,795,658.50	286,897,077.71	165,775,906.31

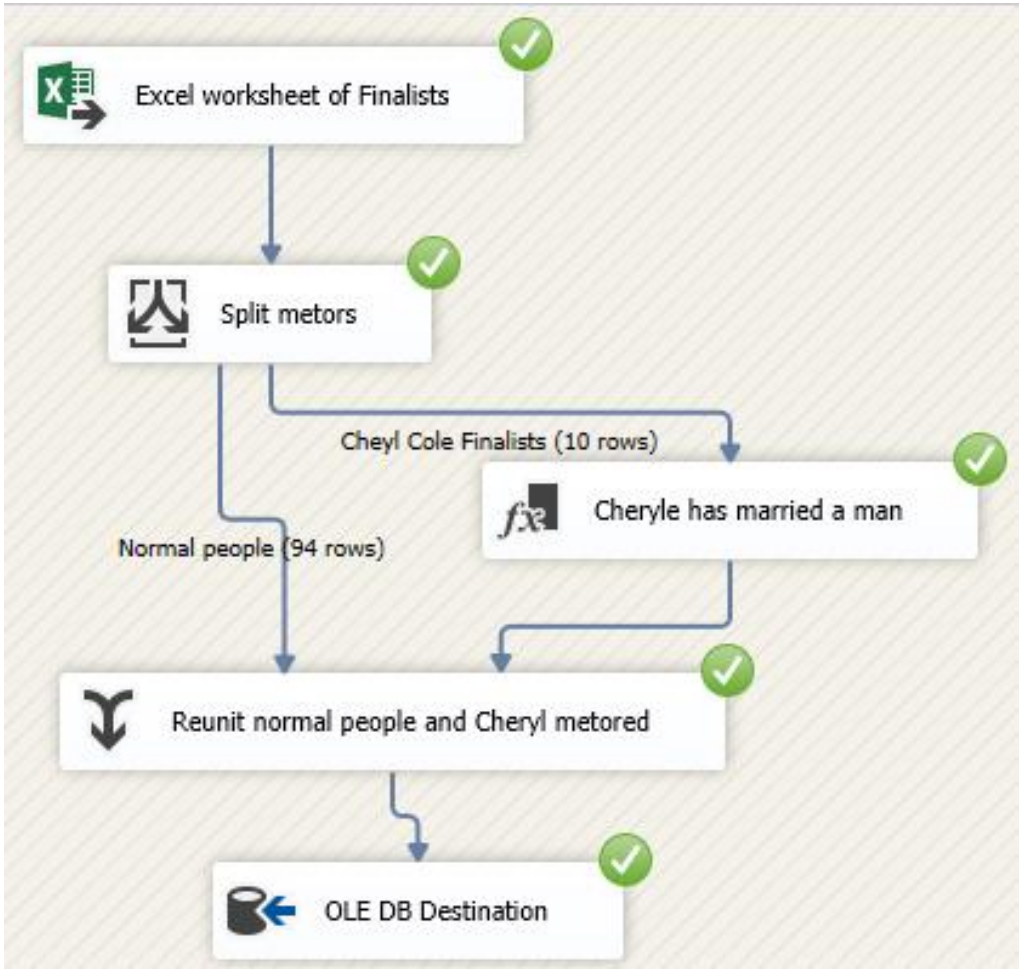
https://github.com/charleseagle/SQL-Server-SSRS

SSIS demos

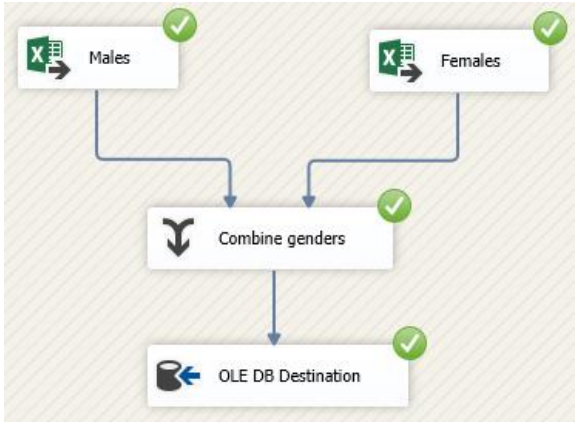
Count series by gender



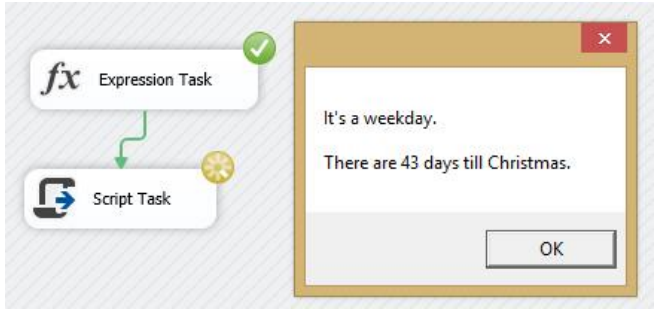
Conditional split



Combine genders



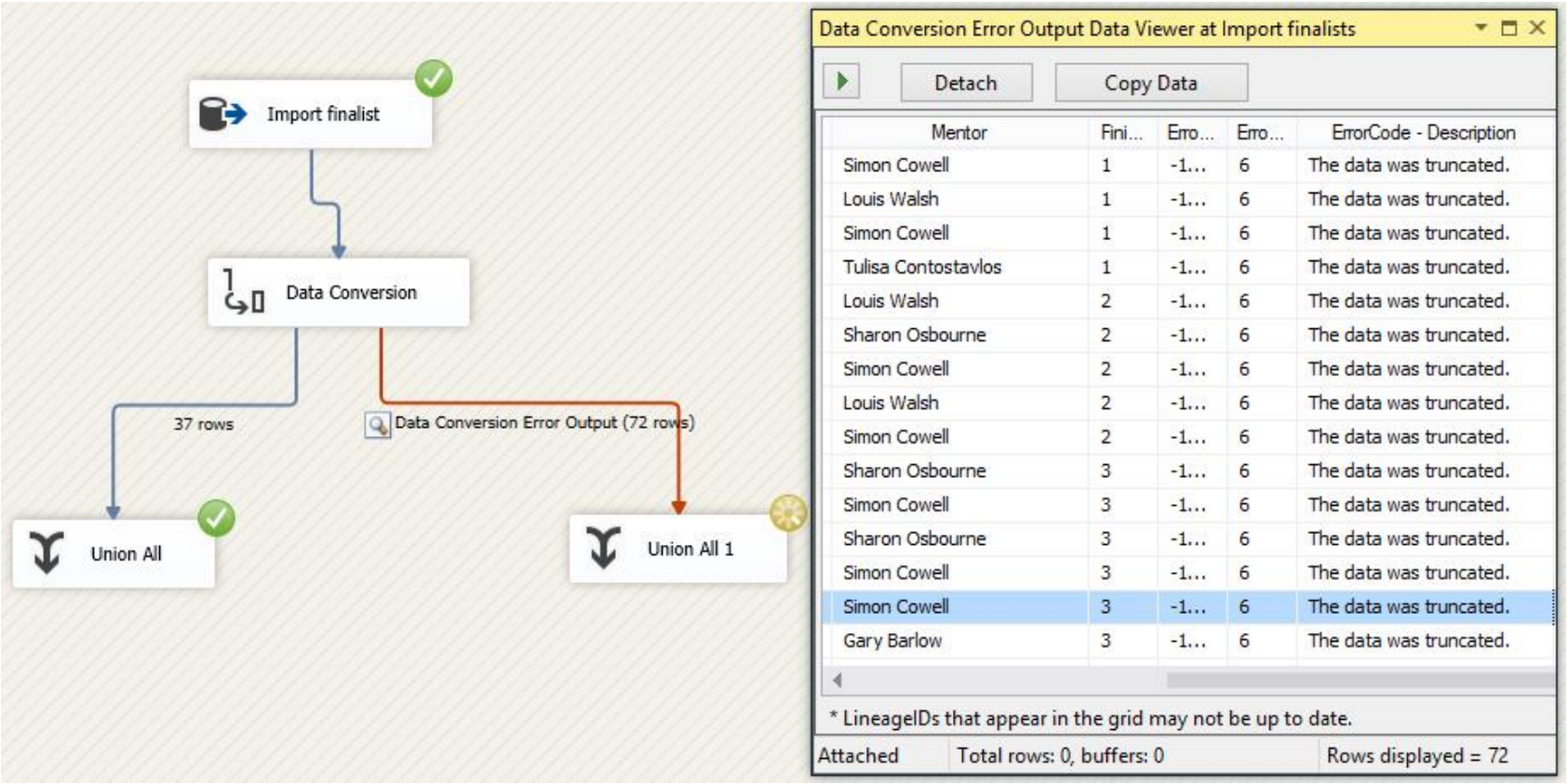
Expression



https://github.com/charleseagle/SQL-Server-SSIS

SSIS demos

Error Handling



Data Conversion Error Output Data Viewer at Import finalists

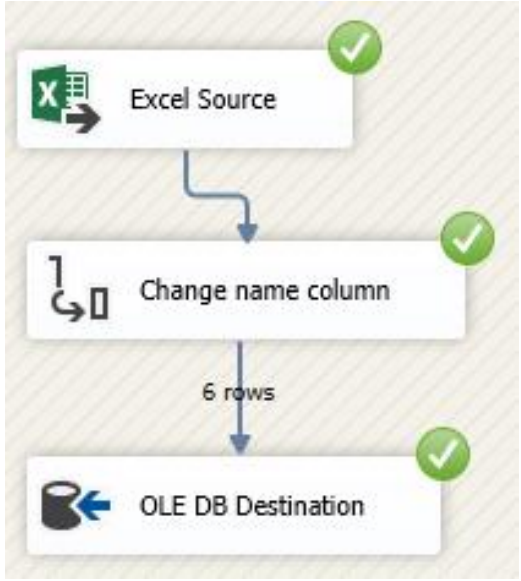
Detach | Copy Data

Mentor	Fini...	Erro...	Erro...	ErrorCode - Description
Simon Cowell	1	-1...	6	The data was truncated.
Louis Walsh	1	-1...	6	The data was truncated.
Simon Cowell	1	-1...	6	The data was truncated.
Tulisa Contostavlos	1	-1...	6	The data was truncated.
Louis Walsh	2	-1...	6	The data was truncated.
Sharon Osbourne	2	-1...	6	The data was truncated.
Simon Cowell	2	-1...	6	The data was truncated.
Louis Walsh	2	-1...	6	The data was truncated.
Simon Cowell	2	-1...	6	The data was truncated.
Sharon Osbourne	3	-1...	6	The data was truncated.
Simon Cowell	3	-1...	6	The data was truncated.
Sharon Osbourne	3	-1...	6	The data was truncated.
Simon Cowell	3	-1...	6	The data was truncated.
Simon Cowell	3	-1...	6	The data was truncated.
Gary Barlow	3	-1...	6	The data was truncated.

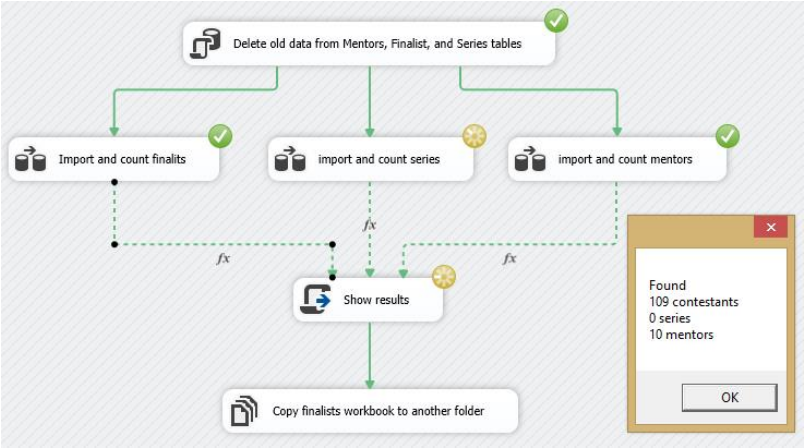
* LineageIDs that appear in the grid may not be up to date.

Attached | Total rows: 0, buffers: 0 | Rows displayed = 72

Data Type Conversion



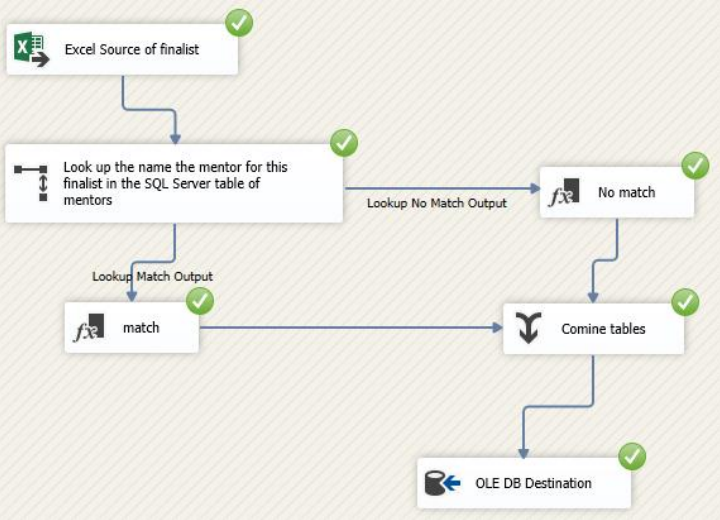
Expression Constrains



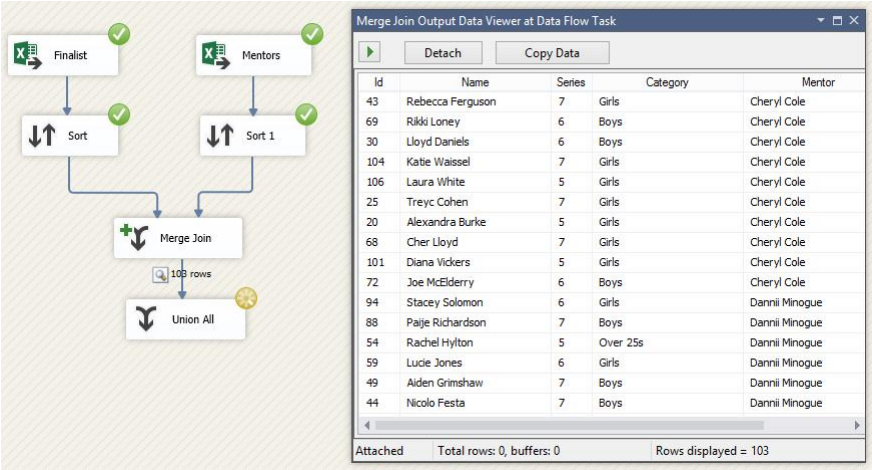
https://github.com/charleseagle/SQL-Server-SSIS

SSIS demos

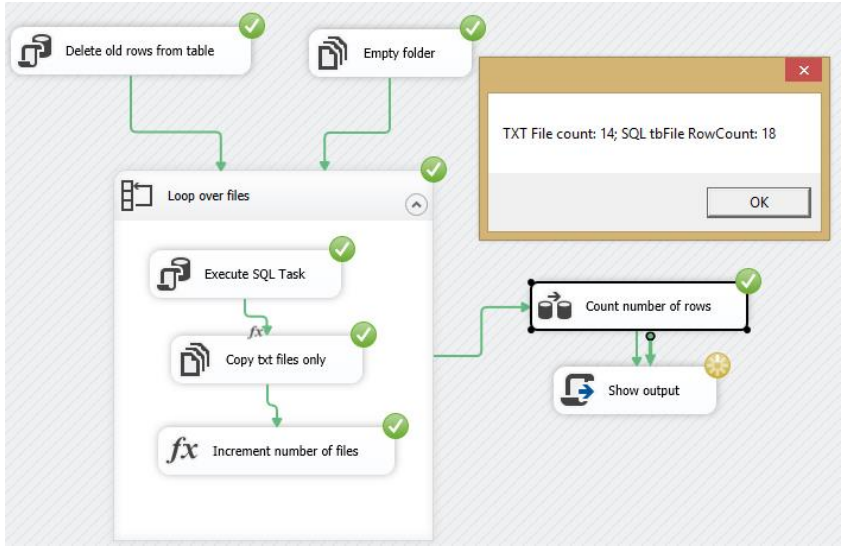
Lookup Transform



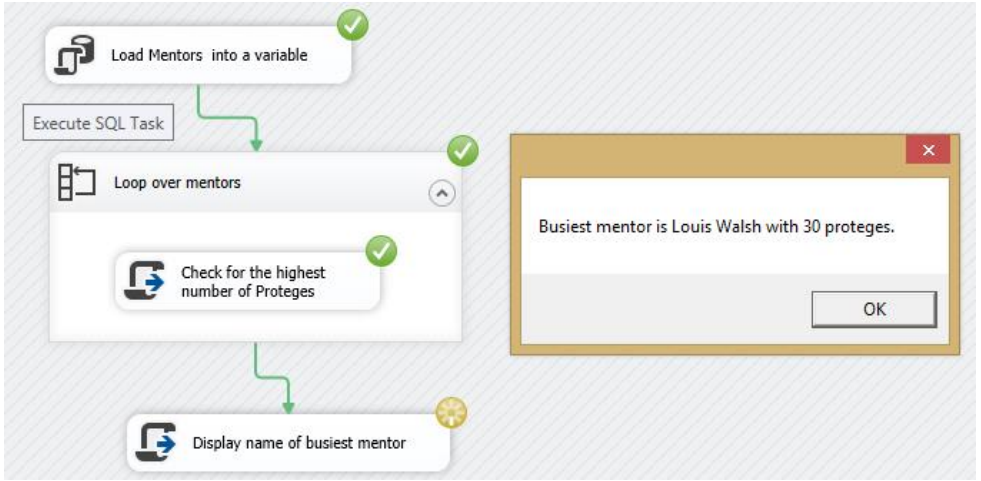
Merge join



Looping over files



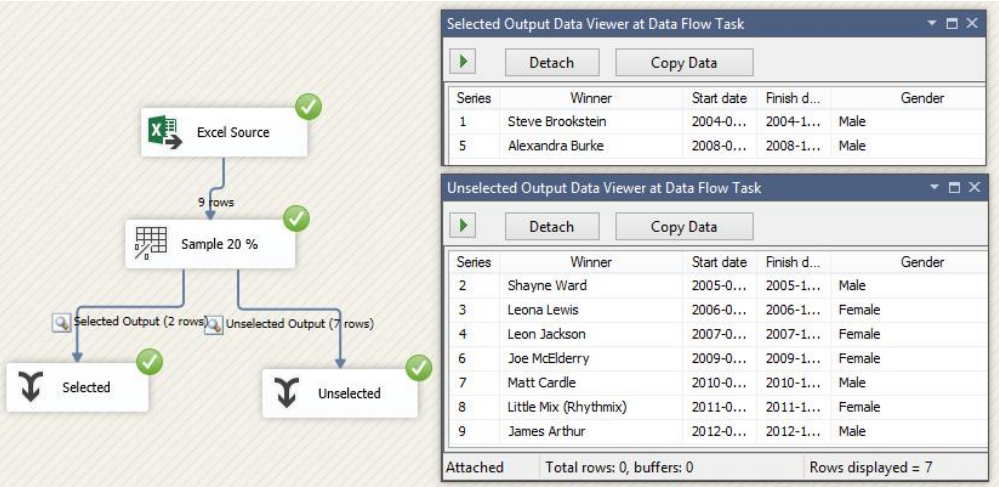
Foreach loop



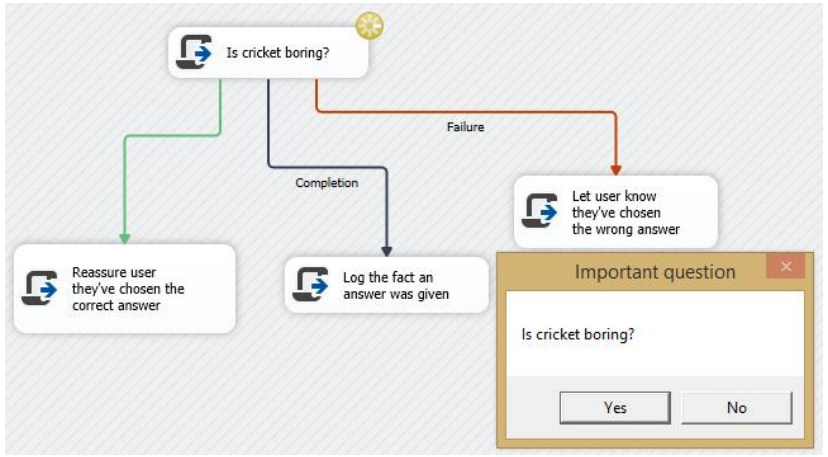
https://github.com/charleseagle/SQL-Server-SSIS

SSIS demos

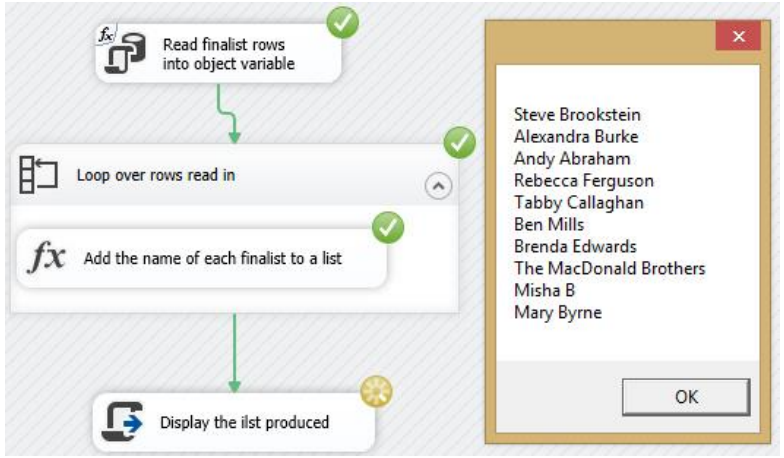
Percentage sampling



Multiple choice

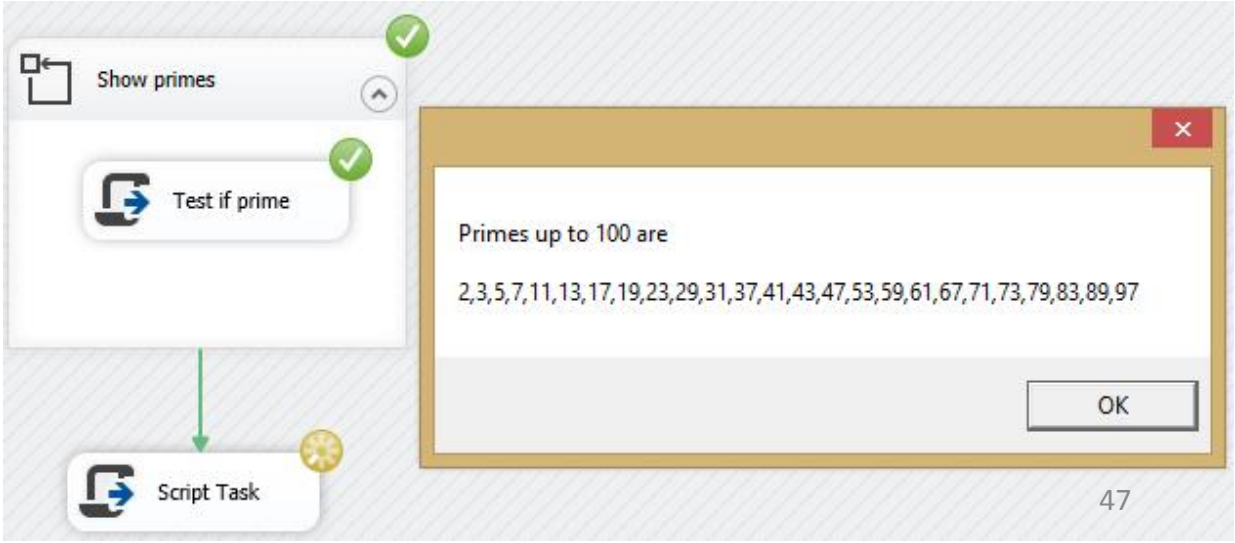


Parameters deployment



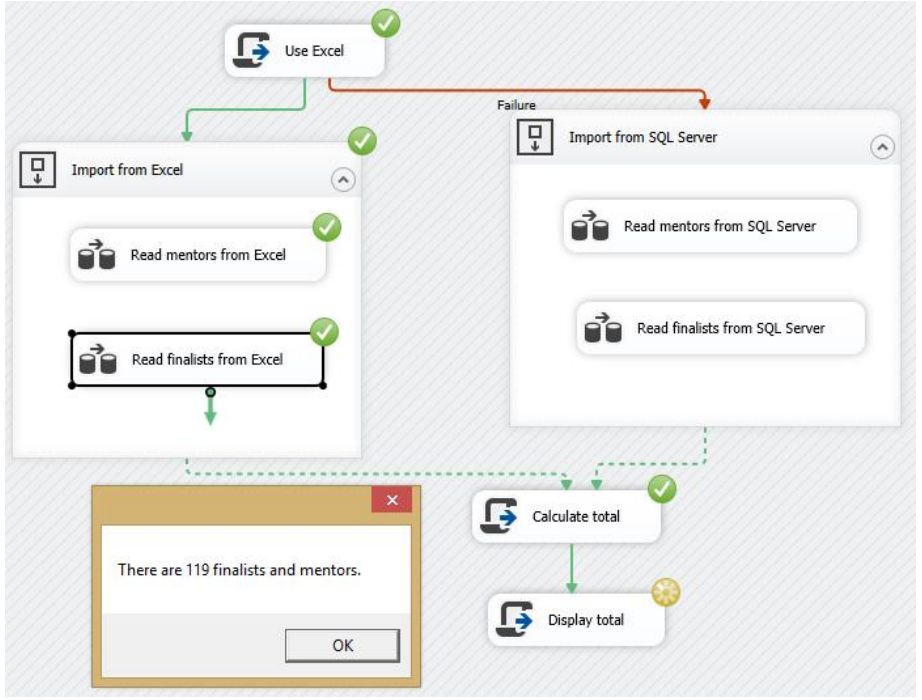
https://github.com/charleseagle/SQL-Server-SSIS

For loop

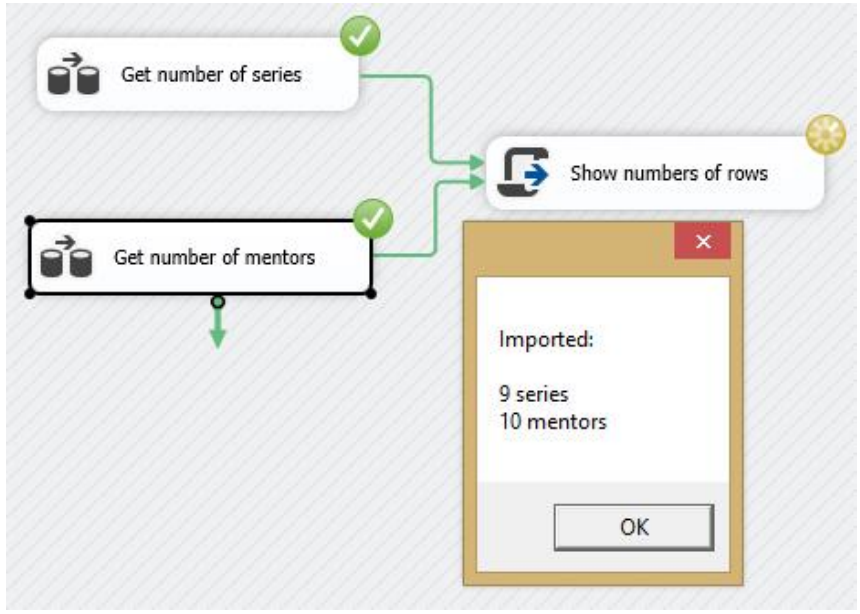


SSIS demos

Script task using C#



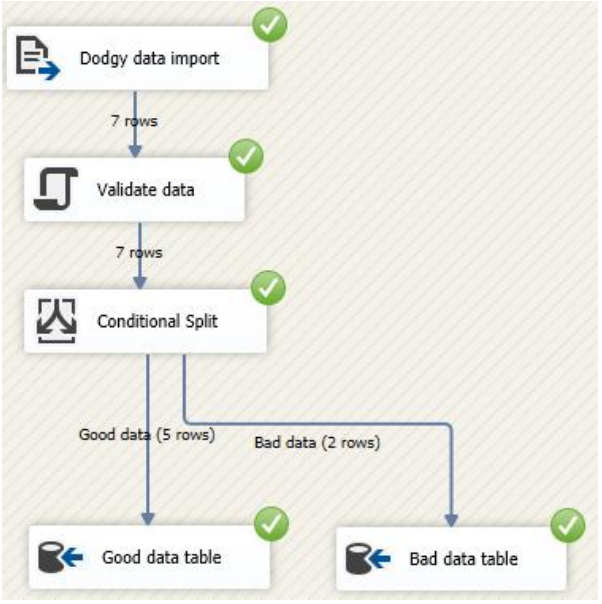
Show row number



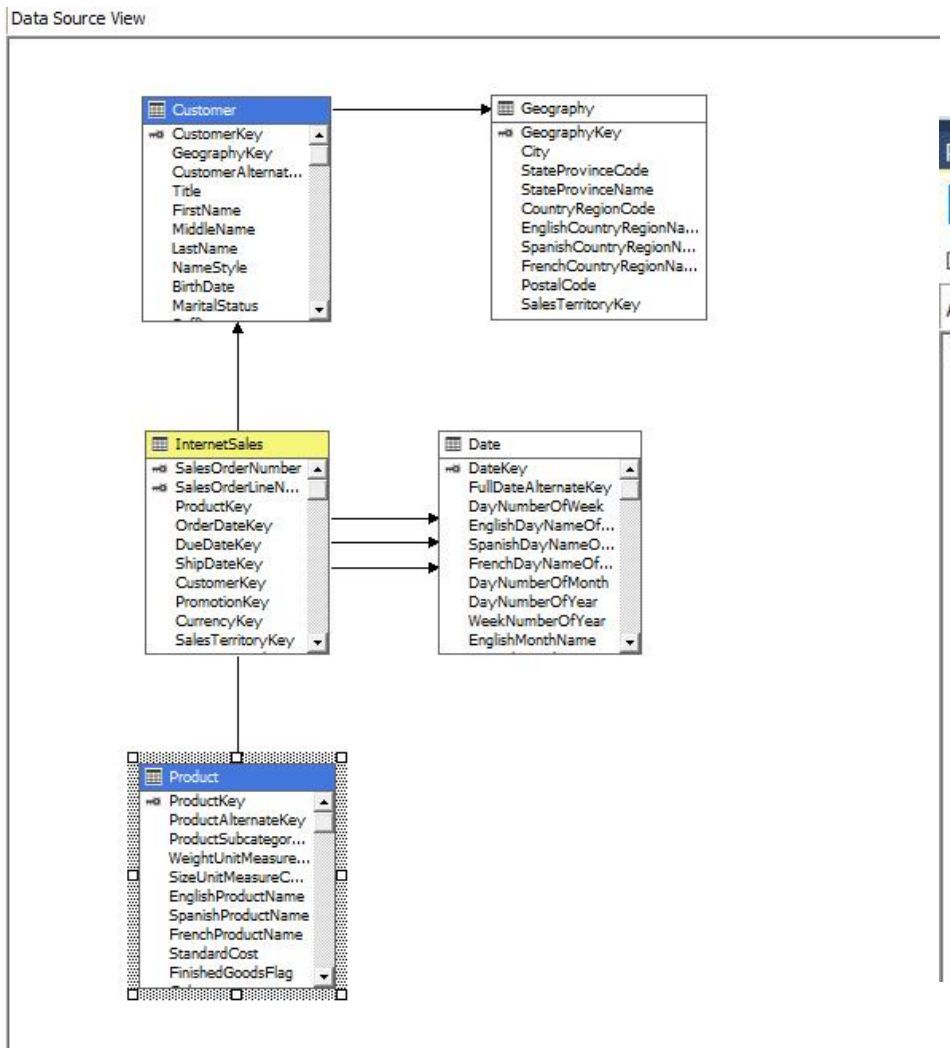
Variables

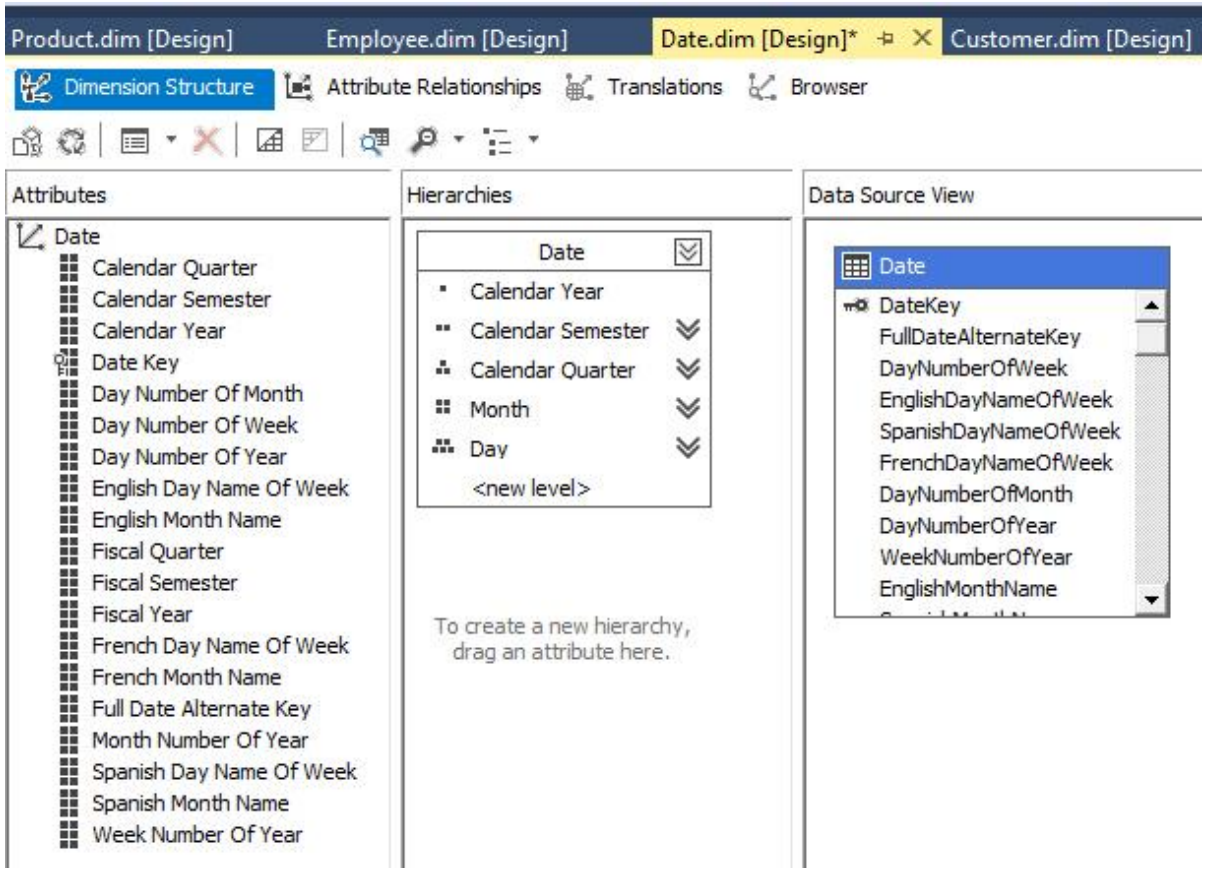
https://github.com/charleseagle/SQL-Server-SSIS

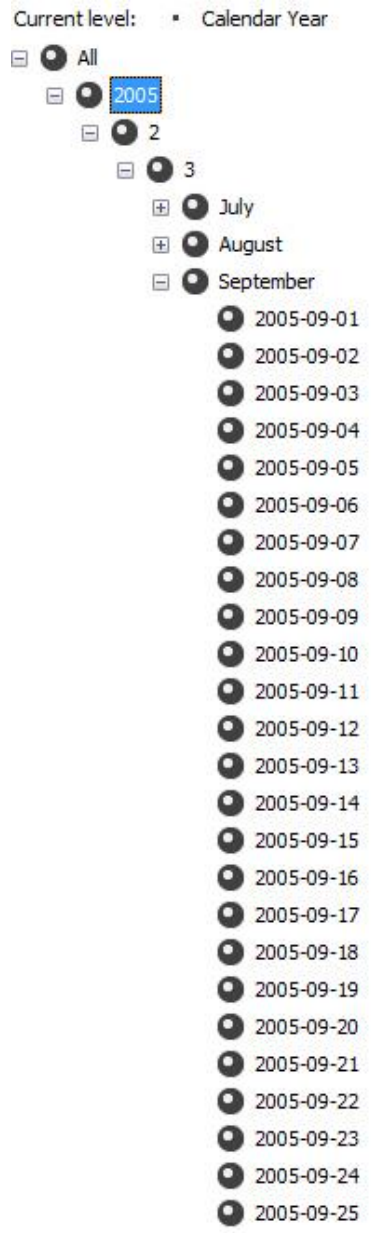
Script component



SSAS demo (CUBE)







https://github.com/charleseagle/SQL-Server-SSAS

Excel BI demos

Key Performance Indicators (KPIs):

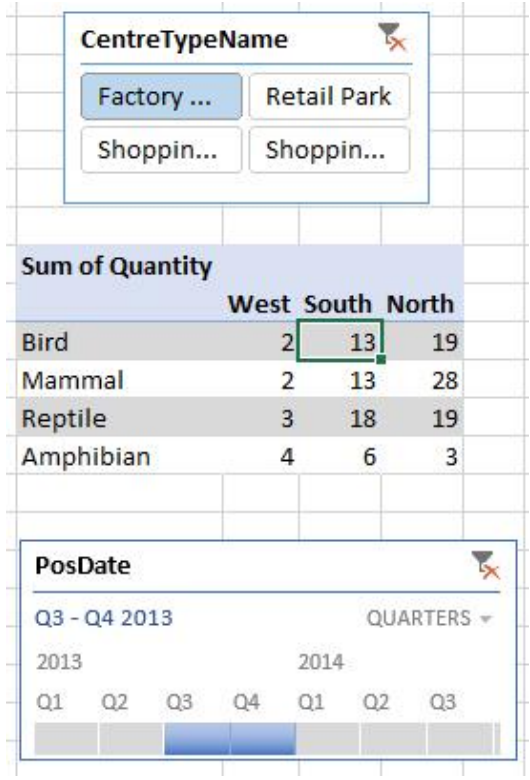
Row Labels	Ratio	Ratio Goal	Ratio Status
2014	100.80 %	100.00 %	
Q1	105.27 %	100.00 %	
February	103.43 %	100.00 %	
January	104.64 %	100.00 %	
March	107.38 %	100.00 %	
Q2	104.37 %	100.00 %	
April	93.73 %	100.00 %	
June	107.19 %	100.00 %	
May	112.95 %	100.00 %	
Q3	98.83 %	100.00 %	
August	105.13 %	100.00 %	
July	91.82 %	100.00 %	
September	101.17 %	100.00 %	
Q4	95.28 %	100.00 %	
December	89.82 %	100.00 %	
November	96.25 %	100.00 %	
October	99.76 %	100.00 %	
Grand Total	100.80 %	100.00 %	

Hierarchy:

Classification	Frog						
Sum of Quantity	Column Labels						
	2013				2013 Total	2014	Grand Total
Row Labels	Q1	Q2	Q3	Q4			
East	15	13	13	14	55	83	138
East Anglia	4	2	5	7	18	17	35
Barhill			5		5	2	7
Cambridge				1	1	10	11
Ipswich	2				2	1	3
Lowestoft	2			1	3		3
Norwich		1		4	5	3	8
Peterborough		1		1	2	1	3
East Midlands	11	11	8	7	37	66	103
North	51	59	68	70	248	289	537
South	70	66	88	83	307	220	527
West	27	28	43	38	136	98	234
Grand Total	163	166	212	205	746	690	1436

PowerPivot:

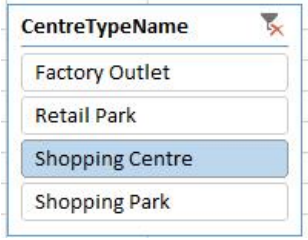
Filter:



https://github.com/charleseagle/Excel-Business-Intelligence-including-Power-Pivot-Power-VIew-Power-Query-Power-Map

Calculated Function DAX (Data Analysis Expressions):

	Column Labels									
	Amphibian		Bird		Mammal		Reptile		Total Right	Total Qty
Row Labels	Right	Qty	Right	Qty	Right	Qty	Right	Qty		
East	79	79	262	262	428	428	264	264	1033	1033
North	332	332	1221	1221	1714	1714	1191	1191	4458	4458
South		404		1400		1972		1426		5202
West	167	167	622	622	952	952	609	609	2350	2350
Grand Total	578	982	2105	3505	3094	5066	2064	3490	7841	13043

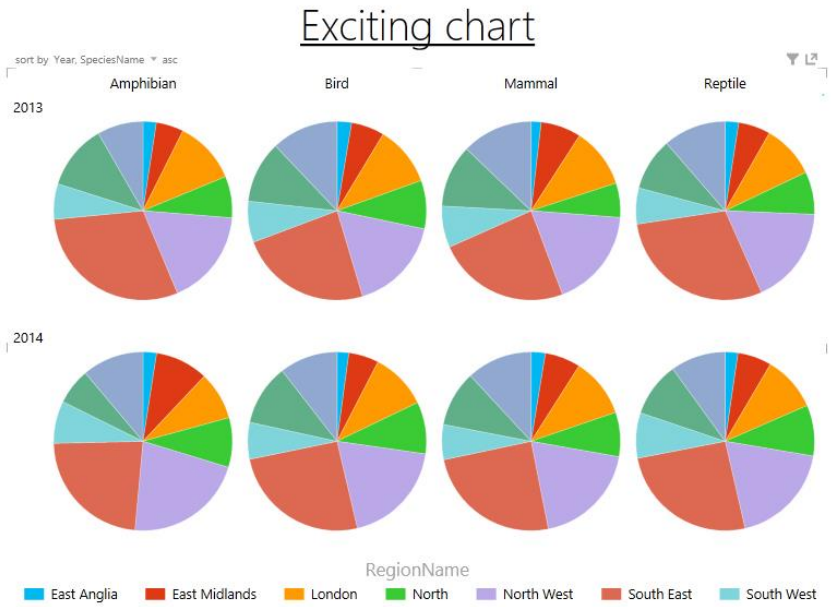


Calculated Columns:

Sum of Amount	Column Labels			
Row Labels	High	Low	Medium	Grand Total
Cat		6130.54	9468.02	15598.56
Possum			9468.02	9468.02
Snowy		6130.54		6130.54
Crocodile	7370.925		6732.33	14103.255
Crocky	7370.925		6732.33	14103.255
Frog			10985.3305	10985.3305
Fred			10985.3305	10985.3305
Owl	230	4955.591	11848.5	17034.091
Speccles	230		11848.5	12078.5
Wol		4955.591		4955.591
Penguin	21779.1873		6052.2	27831.3873
Pingu	8400.48		6052.2	14452.68
The Emperor	13378.7073			13378.7073
Sheep		4622.145	12594.41	17216.555
Sean		4622.145	2546.17	7168.315
Woolly			10048.24	10048.24
Snake	27756.3		9210.525	36966.825
Simon	27756.3			27756.3
Slithery			9210.525	9210.525
Tiger			10915.236	10915.236
Tigger			10915.236	10915.236
Grand Total	57136.4123	15708.276	77806.5515	150651.2398

Excel BI demos

Power view:



Power query:

Rank	Country	Population	% of worldpopulation
1	China	1373250000	19
2	India	1280230000	18
3	United States	322257000	4
4	Indonesia	255461700	4
5	Brazil	205209000	3
6	Pakistan	188925000	3
7	Nigeria	182202000	2
8	Bangladesh	159387000	2
9	Russia	146439880	2
10	Japan	126890000	2
11	Mexico	121005000	2
12	Philippines	102370400	1
13	Vietnam	90730000	1
14	Ethiopia	90076012	1
15	Egypt	89873500	1
16	Germany	81197500	1
17	Iran	78795800	1
18	Turkey	77695904	1

Power pivot:

Matrix

Count of StaffId	Column Labels															
Row Labels	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983	1984	1985	1986
1953																
1954	1	1	2	1	2	2		1	1	4	2	1	1	1	1	1
1955		1	1	2	2	3	1	1	3		1	1			5	1
1956			2	3	1		1	1		1	1	1	2	1	3	1
1957			3	4	1		1	1	3		3	2		2	1	3
1958				1	2	2	2	1	1	2	1	1	2	1	1	
1959					1		1			1	2	2	1	2	3	2
1960							3	2	3	1	2	2	1	1	2	1
1961							1	2	2	4	1	4		2	1	1
1962								1	2	4	1	1	2		1	1
1963									2	2		1	1	1		2
1964											2	2	1	4	1	1
1965												2	3	1	1	5
1966												2	1	4	3	1
1967														2	1	1
1968															3	1
1969																3
1970																
1971																

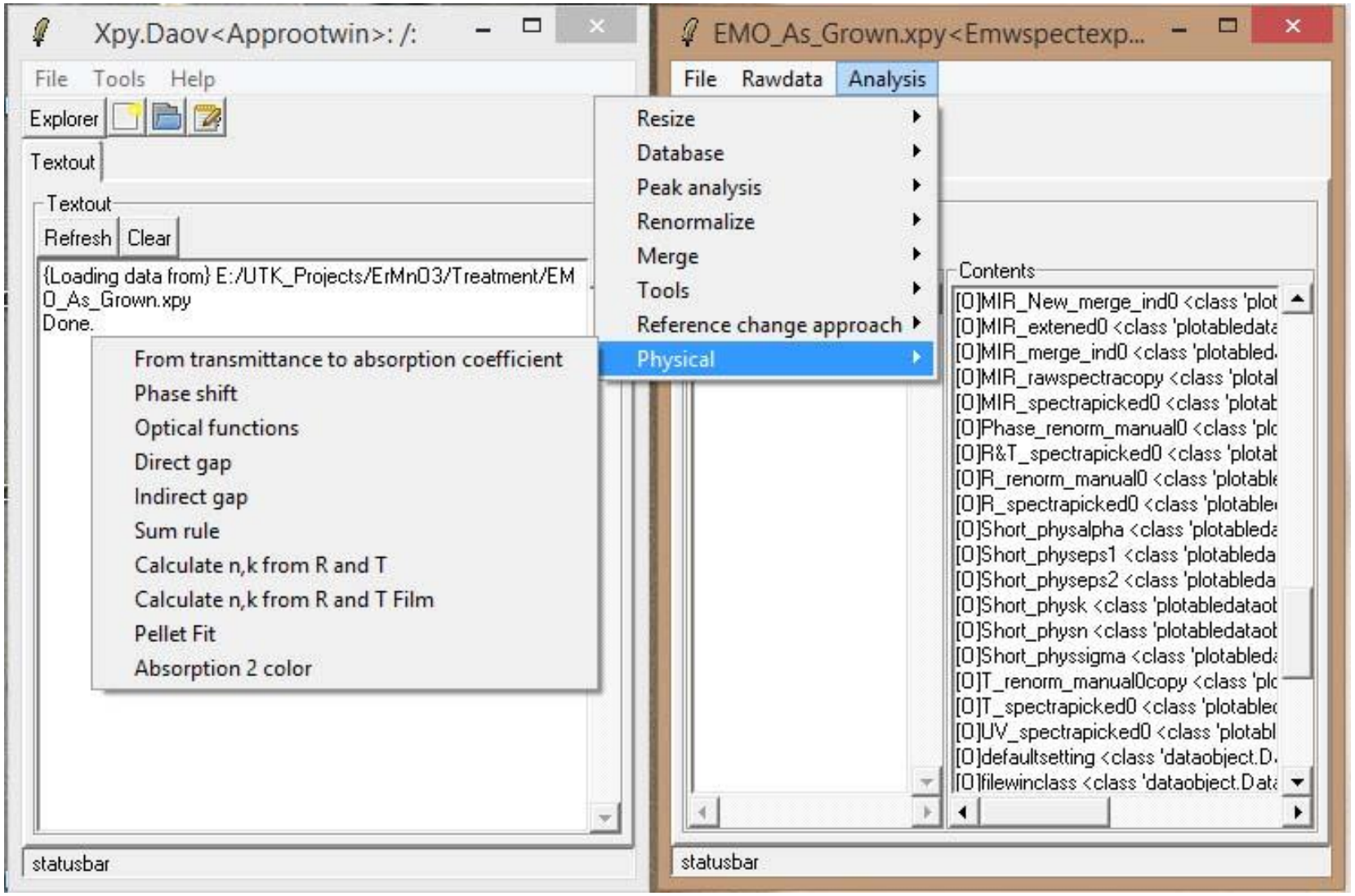
Power map:

https://github.com/charleseagle/Excel-Business-Intelligence-including-Power-Pivot-Power-VIew-Power-Query-Power-Map

Programing demos

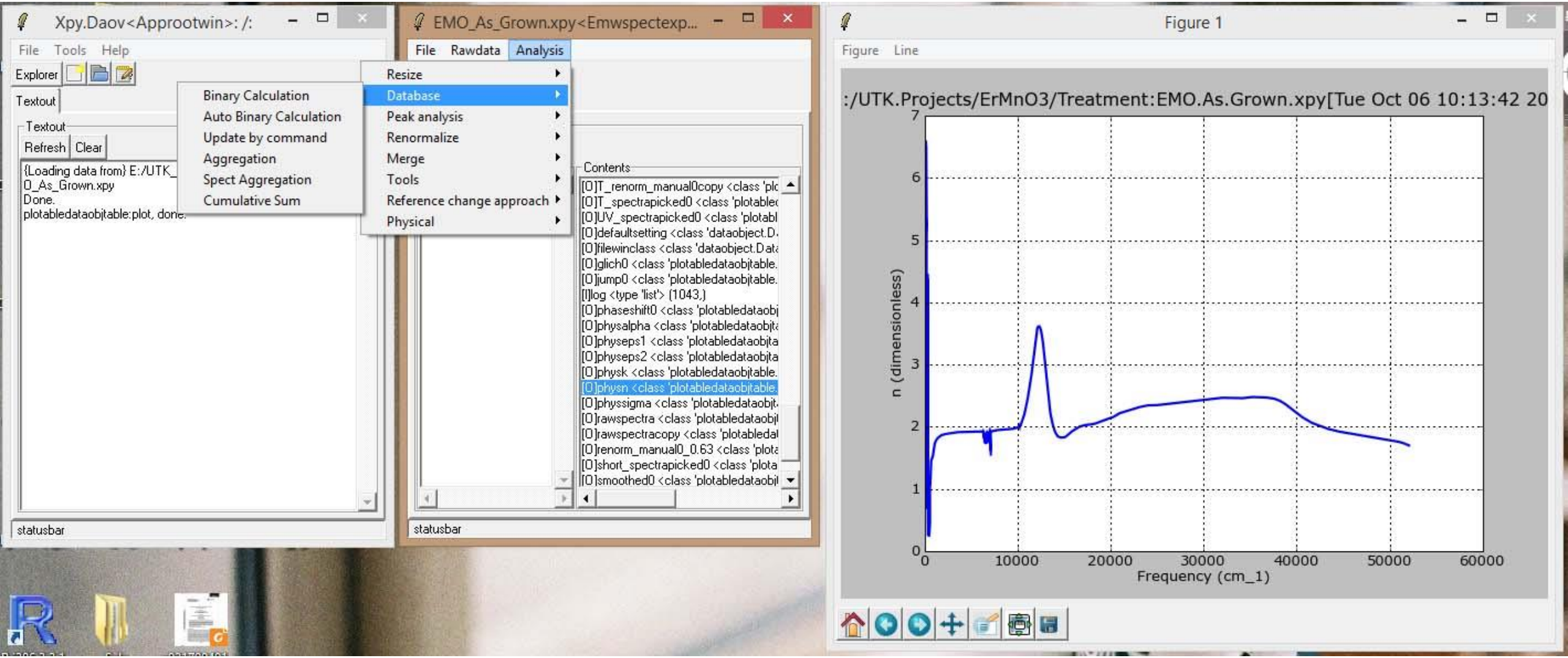
1. Developed software for analyzing data using Python
2. Matlab and Python FTDT simulation
3. Gaussian fit analysis
4. Rewrote Machine Learning codes using Python
5. C# demo for linear regression
6. Labview-controlled photoluminescence system

Analysis software (Python)



https://github.com/charleseagle/Data-Analysis-Software-Python

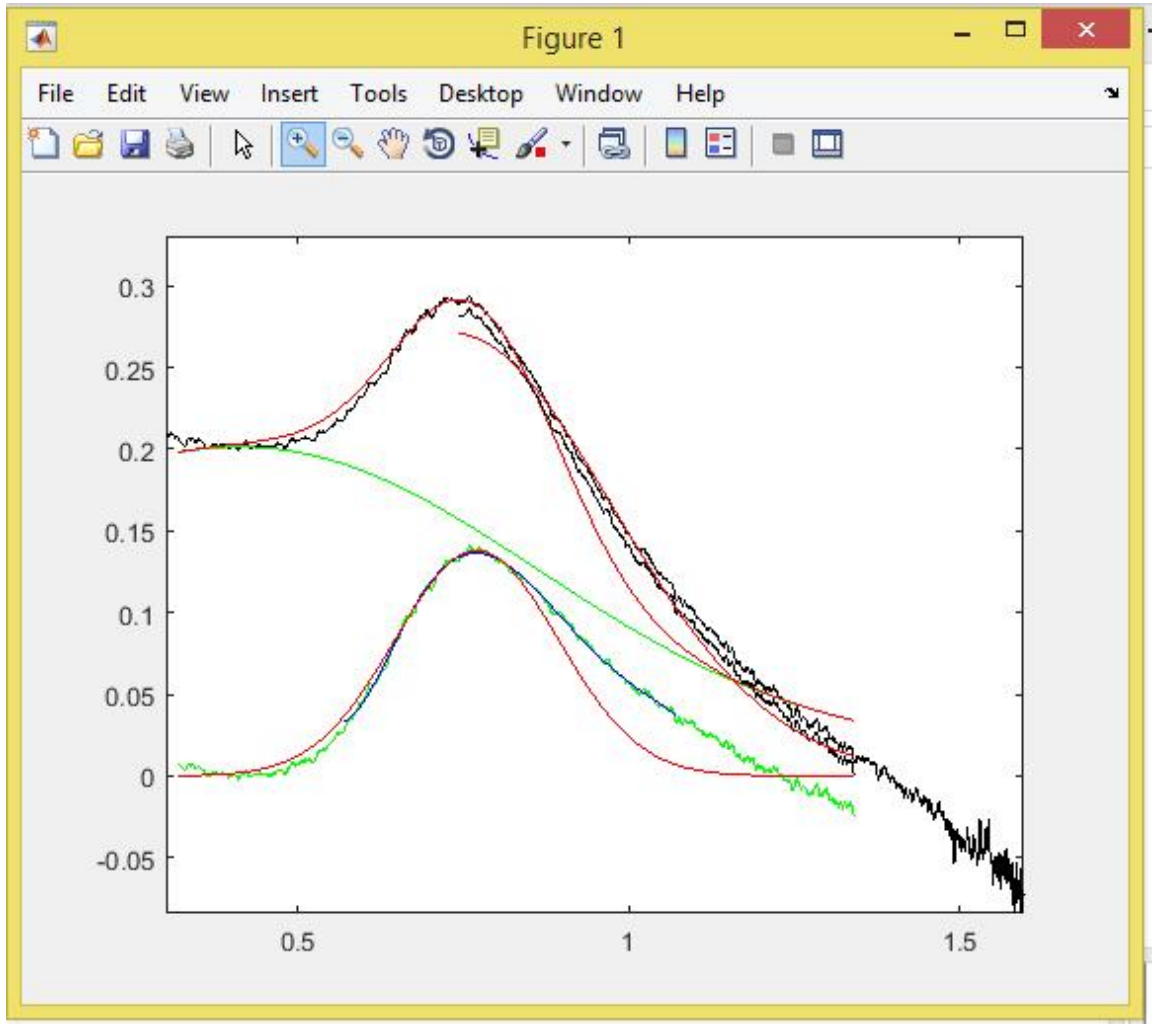
Analysis software (Python)



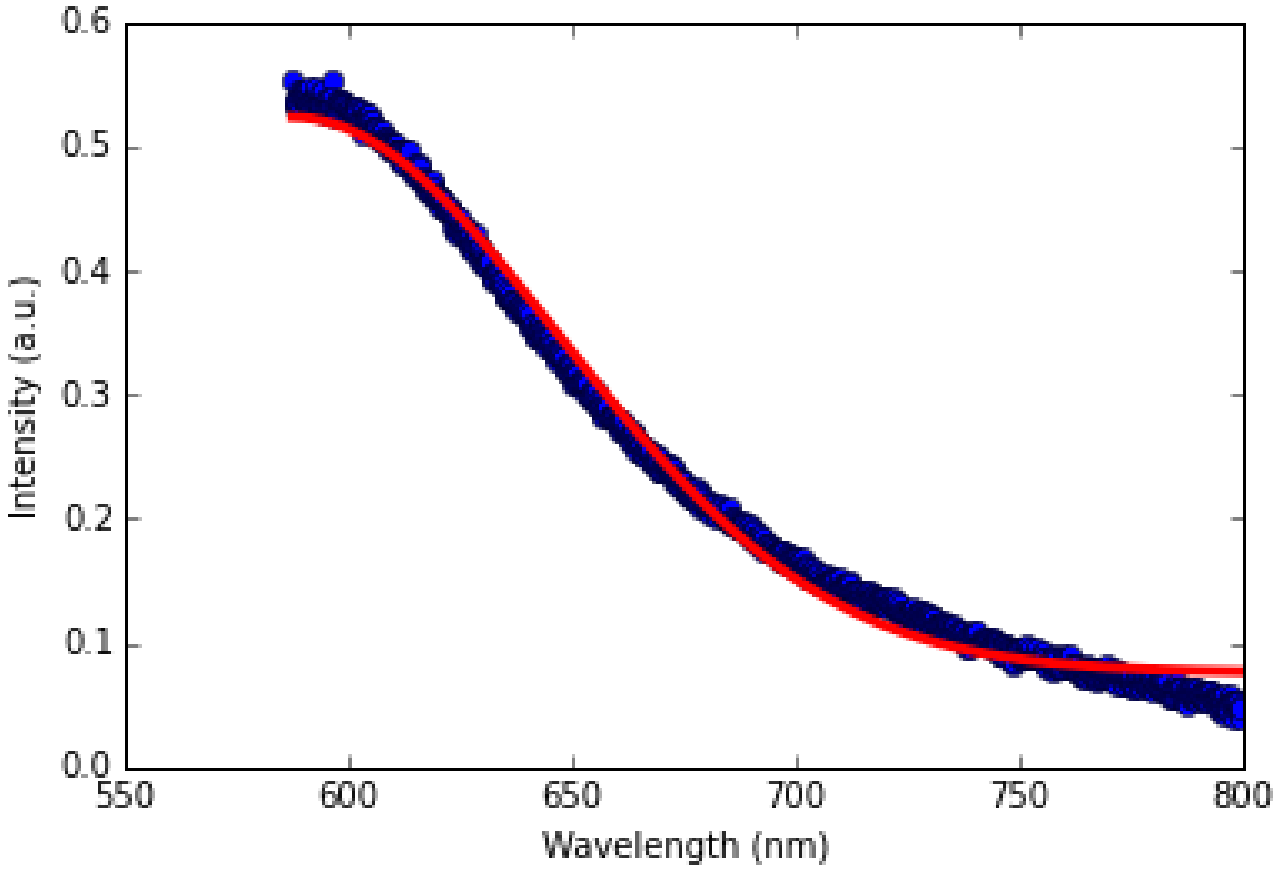
https://github.com/charleseagle/Data-Analysis-Software-Python

Gaussian fit analysis

Matlab demo



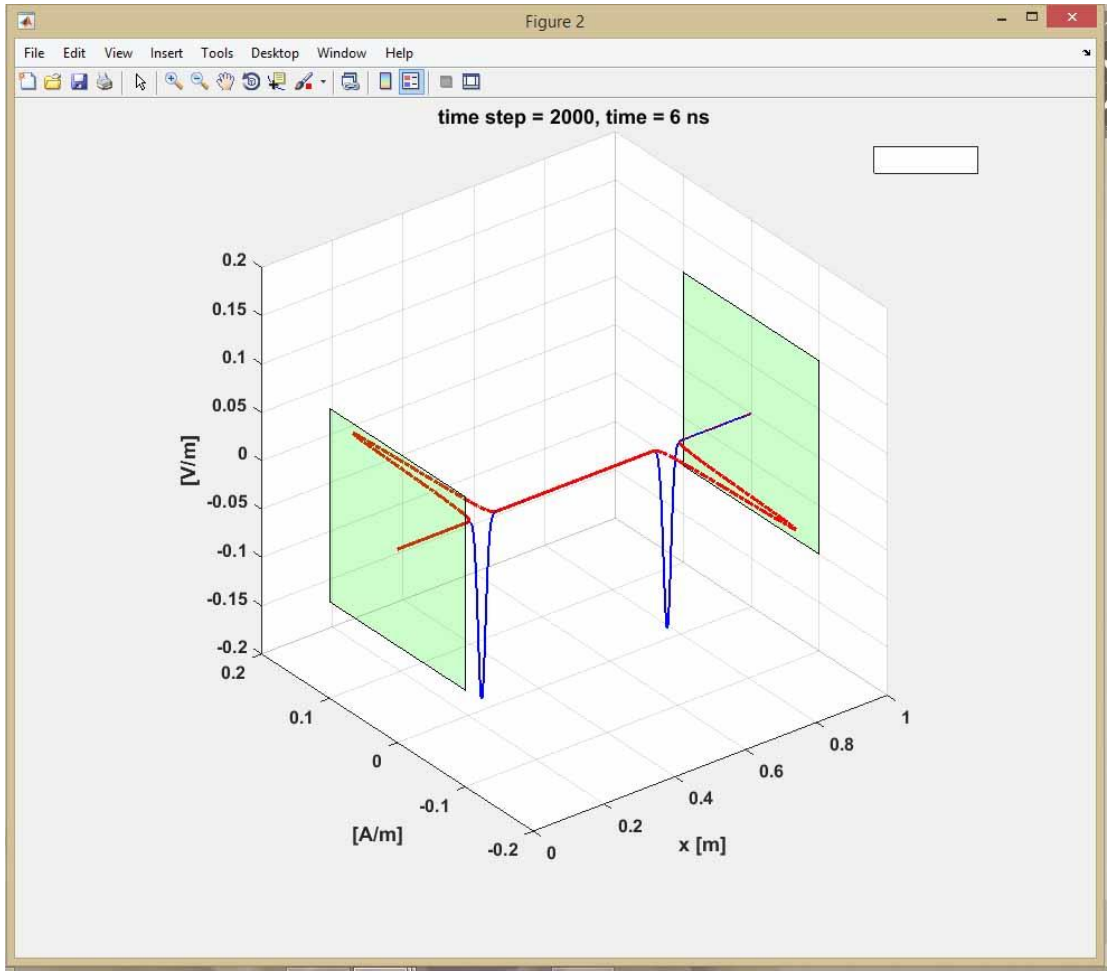
Python demo



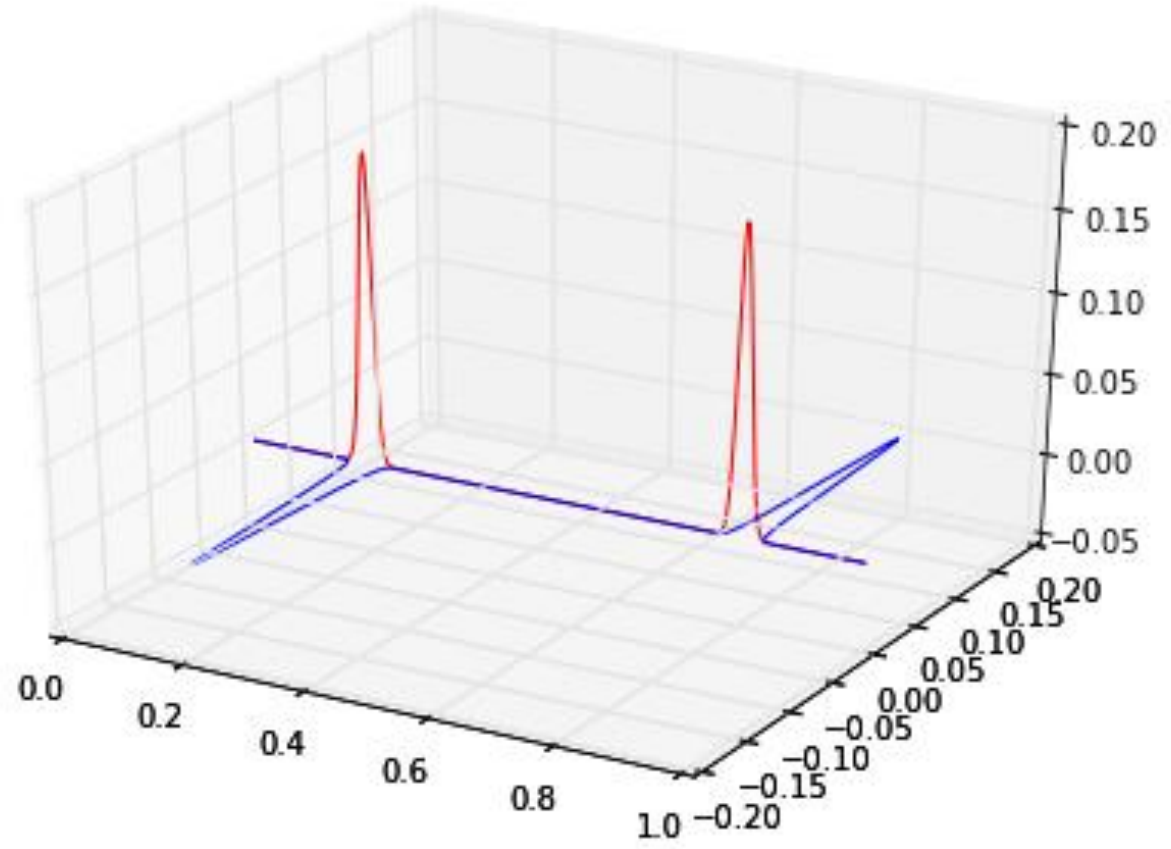
https://github.com/charleseagle/Fitting-analysis-for-spectroscopic-data-using-Python-and-Matlab

FDTD simulation

Matlab demo



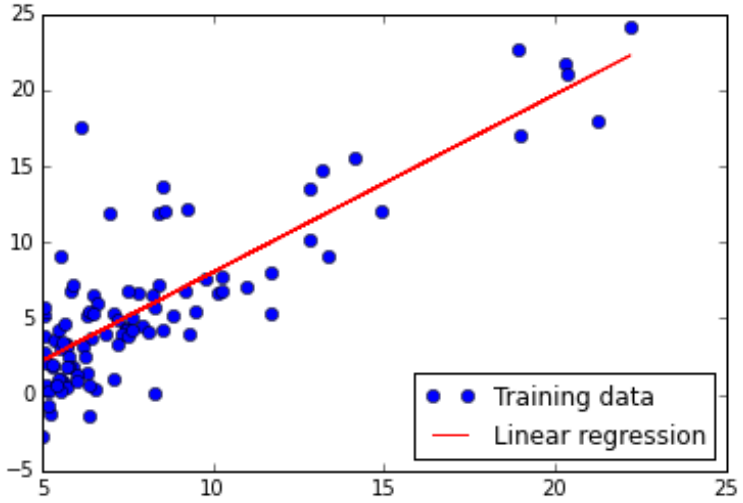
Python demo



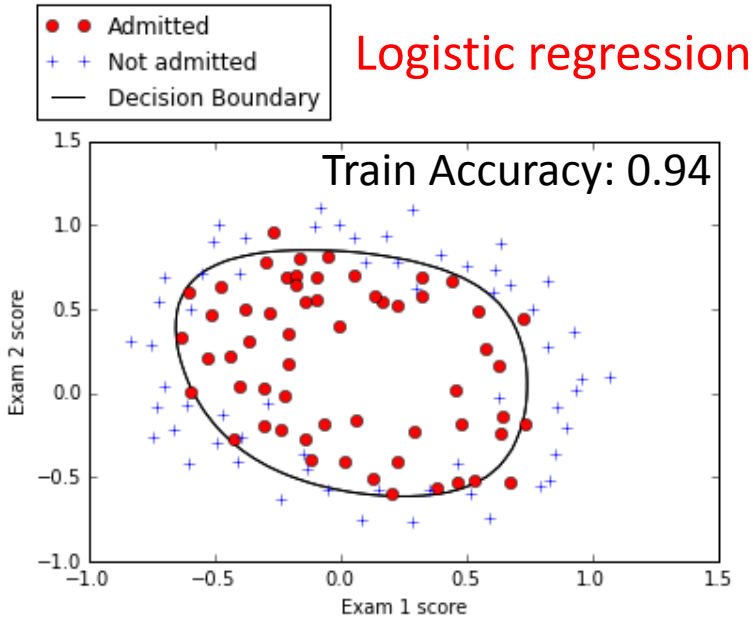
https://github.com/charleseagle/FDTD-simulation-using-Python

Machine Learning (Python)

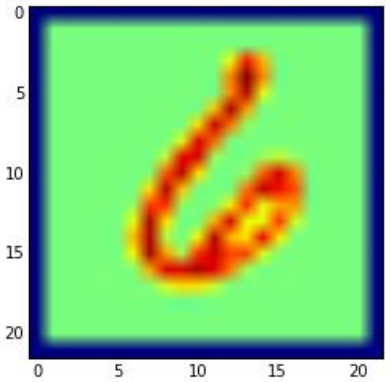
Linear regression



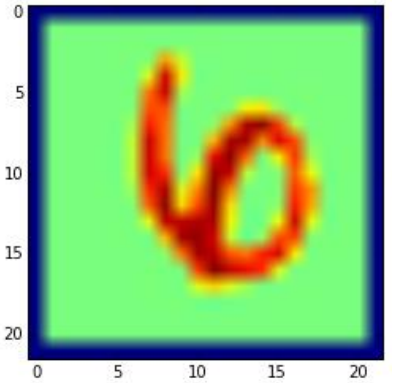
Logistic regression



Neuron network

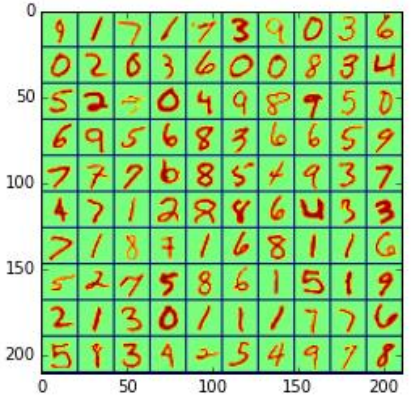


Prediction: 6

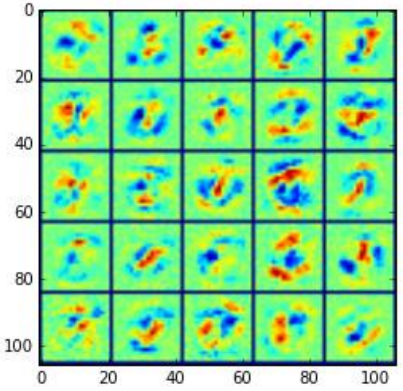


Prediction: 6

Original Data

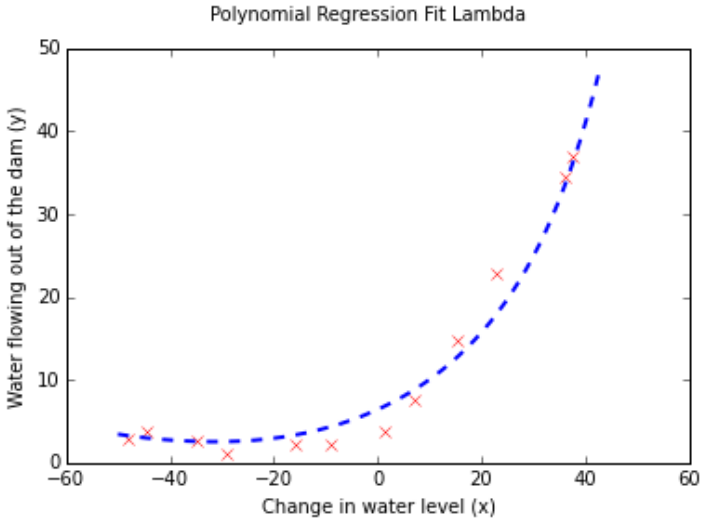


Neural Network

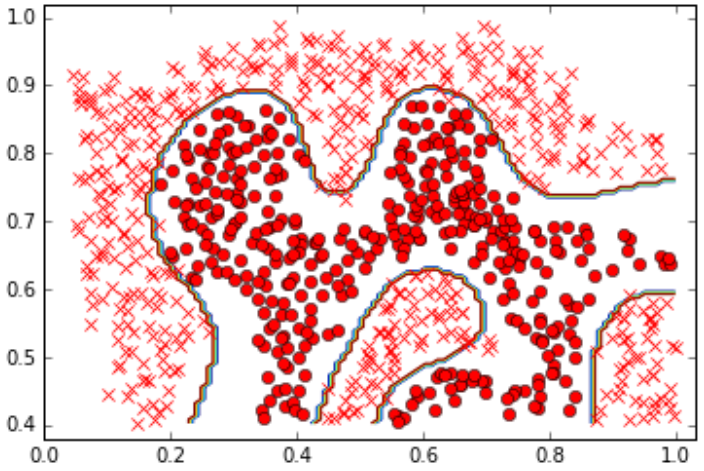


Training Set Accuracy: 0.91

Regularized linear regression



Support vector machines

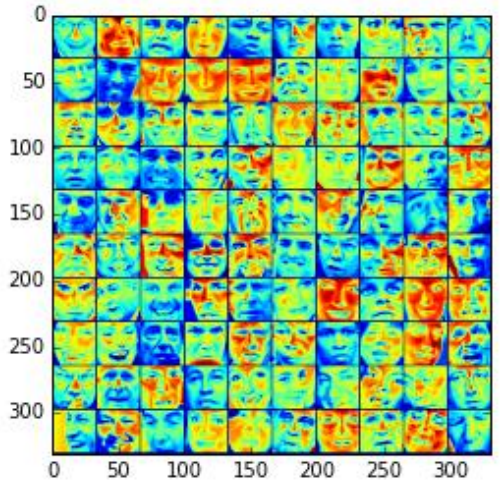


https://github.com/charleseagle/Machine-Learning-Coursera-Python

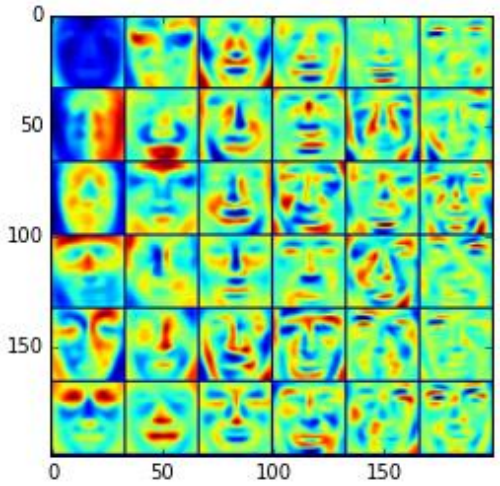
Machine Learning (Python)

Principle component analysis

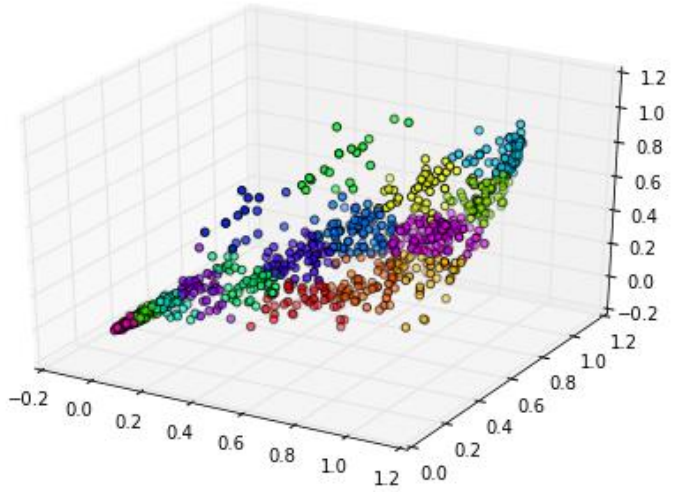
Original data



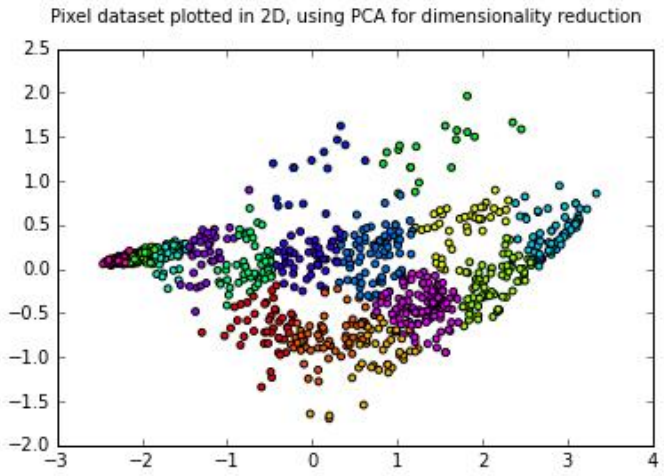
After PCA



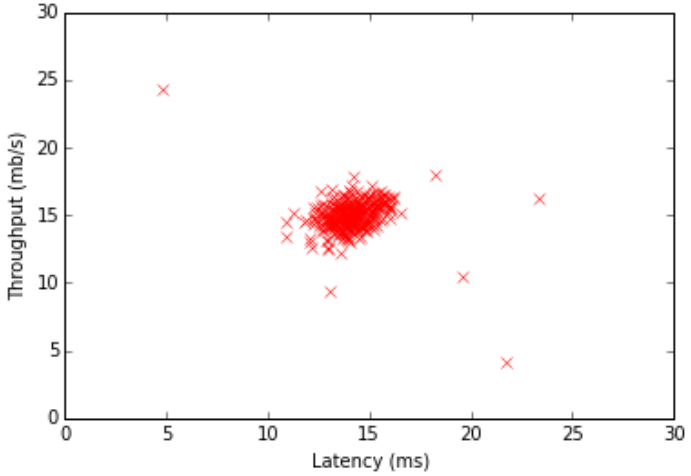
Original data in 3D

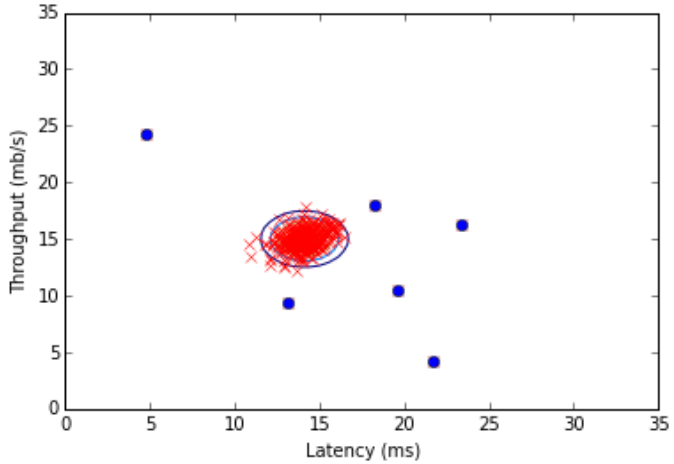


2D visualization using PCA

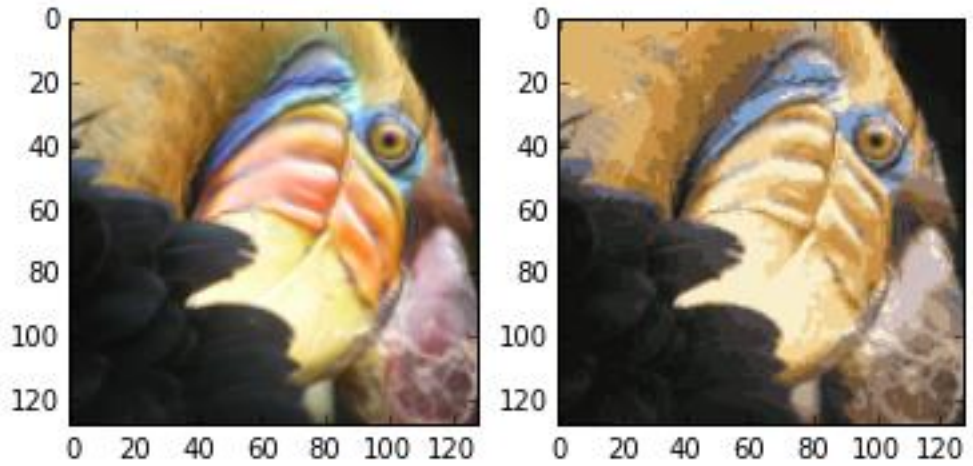


Anomaly detection





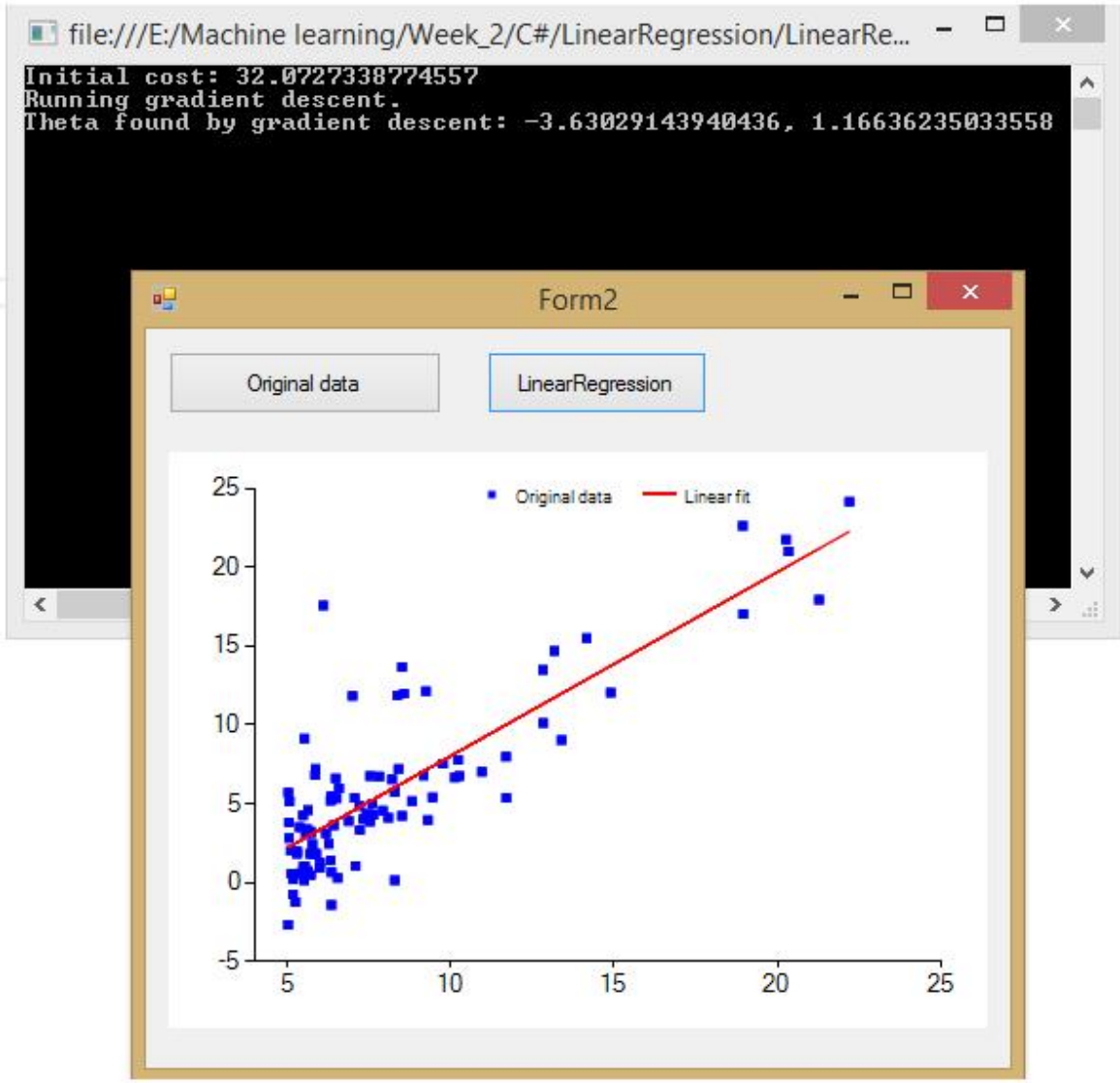
k-mean clustering



https://github.com/charleseagle/Machine-Learning-Coursera-Python

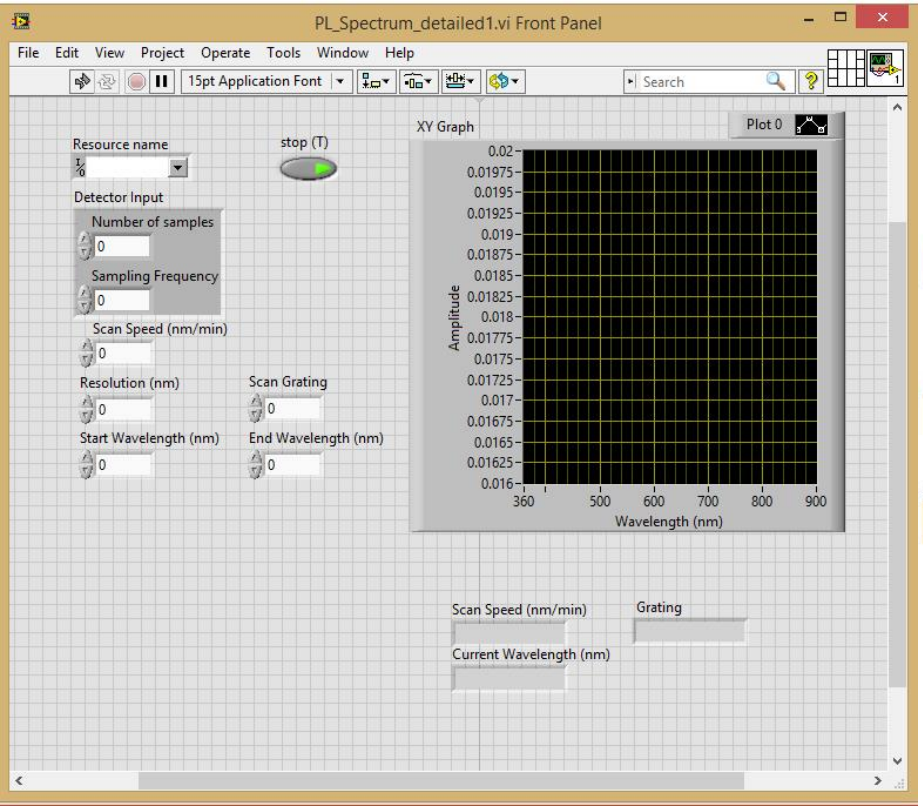
Linear regression (C#)

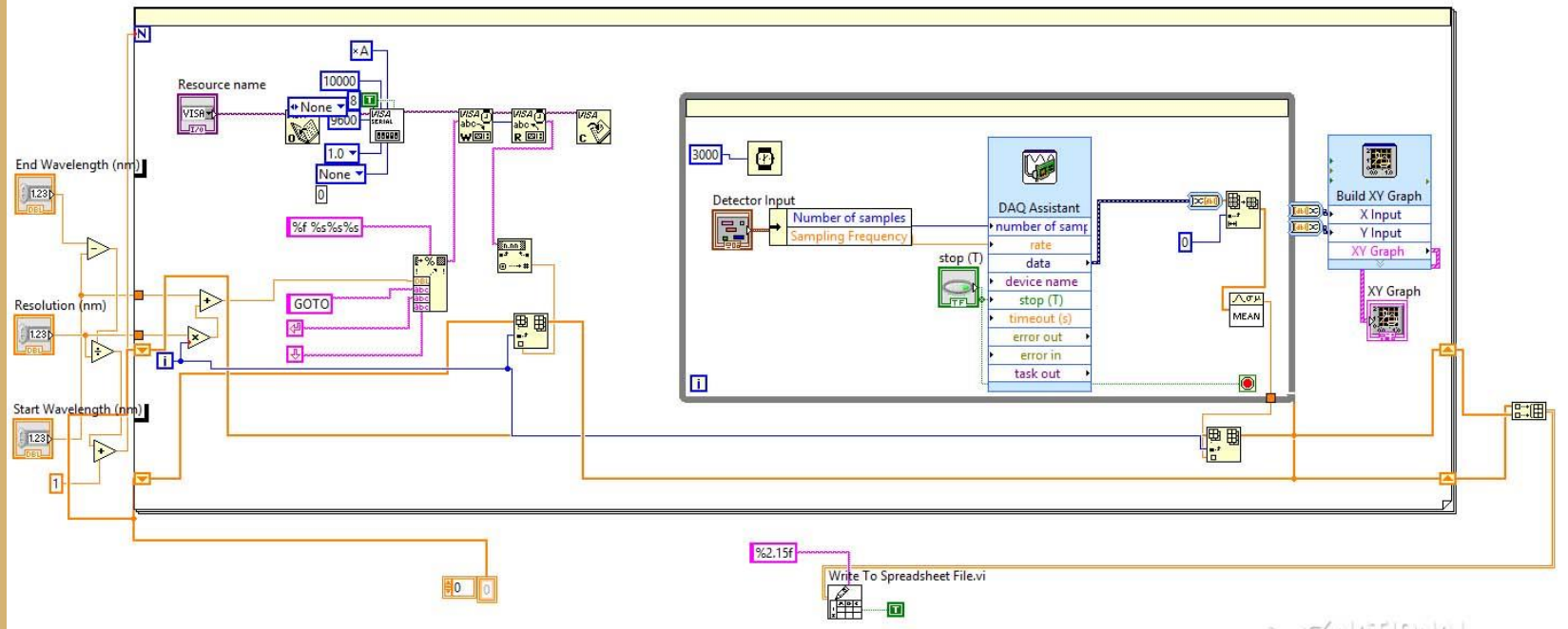
```
99          //Gradient descent for linear regression.
            1 reference
100         public Tuple<double[,], double[,]> Gradient(double[,] X, double[,] y,
101             double[,] theta, double alpha, int num_iters)
102         {
103             double m = y.GetLength(0);
104             double[,] J_history = new double[num_iters,1];
105             for (int i = 0; i < num_iters; i++)
106             {
107                 double[,] temp1 = ArrayMultiply(X, theta);
108                 double[,] temp2 = ArrayMinus(temp1, y);
109                 var temp = new double[X.GetLength(1), X.GetLength(0)];
110                 for (int k = 0; k < X.GetLength(0); k++)
111                 {
112                     for (int j = 0; j < X.GetLength(1); j++)
113                     {
114                         temp[j, k] = X[k, j];
115                     }
116                 }
117                 double[,] temp3 = ArrayMultiply(temp, temp2);
118                 for (int j = 0; j < temp3.GetLength(0); j++)
119                 {
120                     temp3[j, 0] = (alpha / m) * temp3[j, 0];
121                 }
122                 theta = ArrayMinus(theta, temp3);
123                 J_history[i,0] = ComputeCost(X, y, theta);
124             }
125             return new Tuple<double[,], double[,]>(theta, J_history);
126         }
127
128
            2 references
129         public Tuple<double[,], double[,], double, double, double, double[,]> LinearFit()
130         {
131             Program t = new Program();
132             double[,] data;
133             t.FillArray(out data);
134             var X = new double[data.GetLength(0), 2];
135             var y = new double[data.GetLength(0), 1];
```



https://github.com/charleseagle/Machine-Learning-Coursera-C-Sharp

Labview-controlled photoluminescence system





https://github.com/charleseagle/Labview-codes-for-photoluminescence-setup