Assignment #1 : OData and MVC REST services

Introduction

This assignment requires the following three **REST** based services:

- **MVC-SQL**: an **MVC** service retrieving data from an **SQL** Server database
- **OData-SQL**: an **OData** service retrieving data from an **SQL** Server database (same)
- **OData-XML**: an **OData** service retrieving data from several **XML** documents

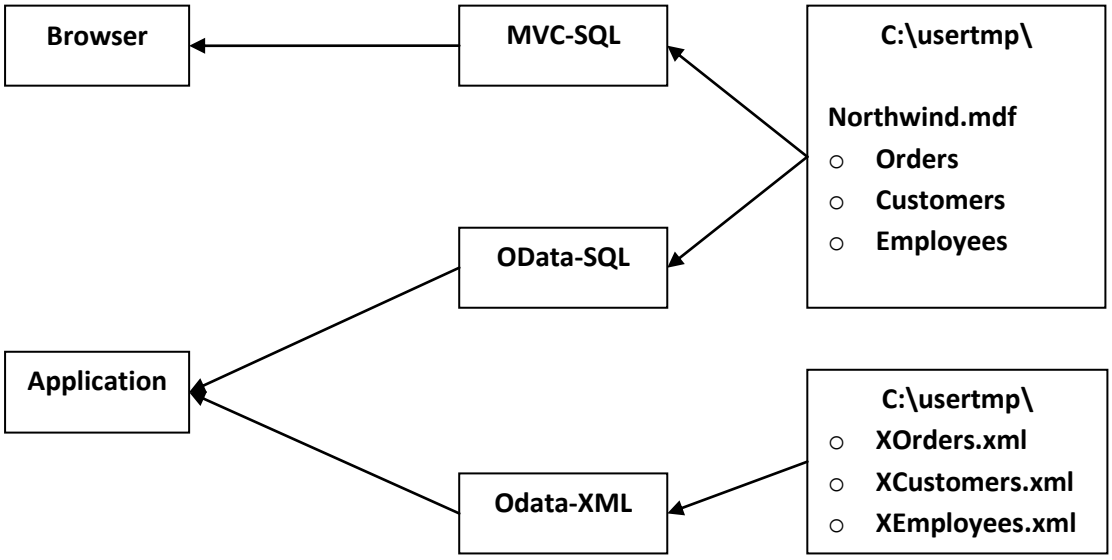
MVC-SQL and **OData-SQL** use three tables - **Orders**, **Customers** and **Employees** -from a slightly modified copy of the public **Northwind SQL** Server database sample, assumed to be located in the **C:\usertmp** local folder.

OData-XML uses trimmed **XML** versions of the same tables – **XOrders.xml**, **XCustomers.xml** and **XEmployees.xml** - also assumed to be located in the **C:\usertmp** local folder.

MVC-SQL returns human readable **HTML** pages containing a read-only server-side paginated grid views, which are directly usable via a **browser**.

OData-SQL and **Odata-XML** return data formatted as **ATOM+XML** or **JSON**, which are mostly usable by other client **applications**.

The following diagram offers a bird's eye view on these three services (arrows indicate the main response data flows):

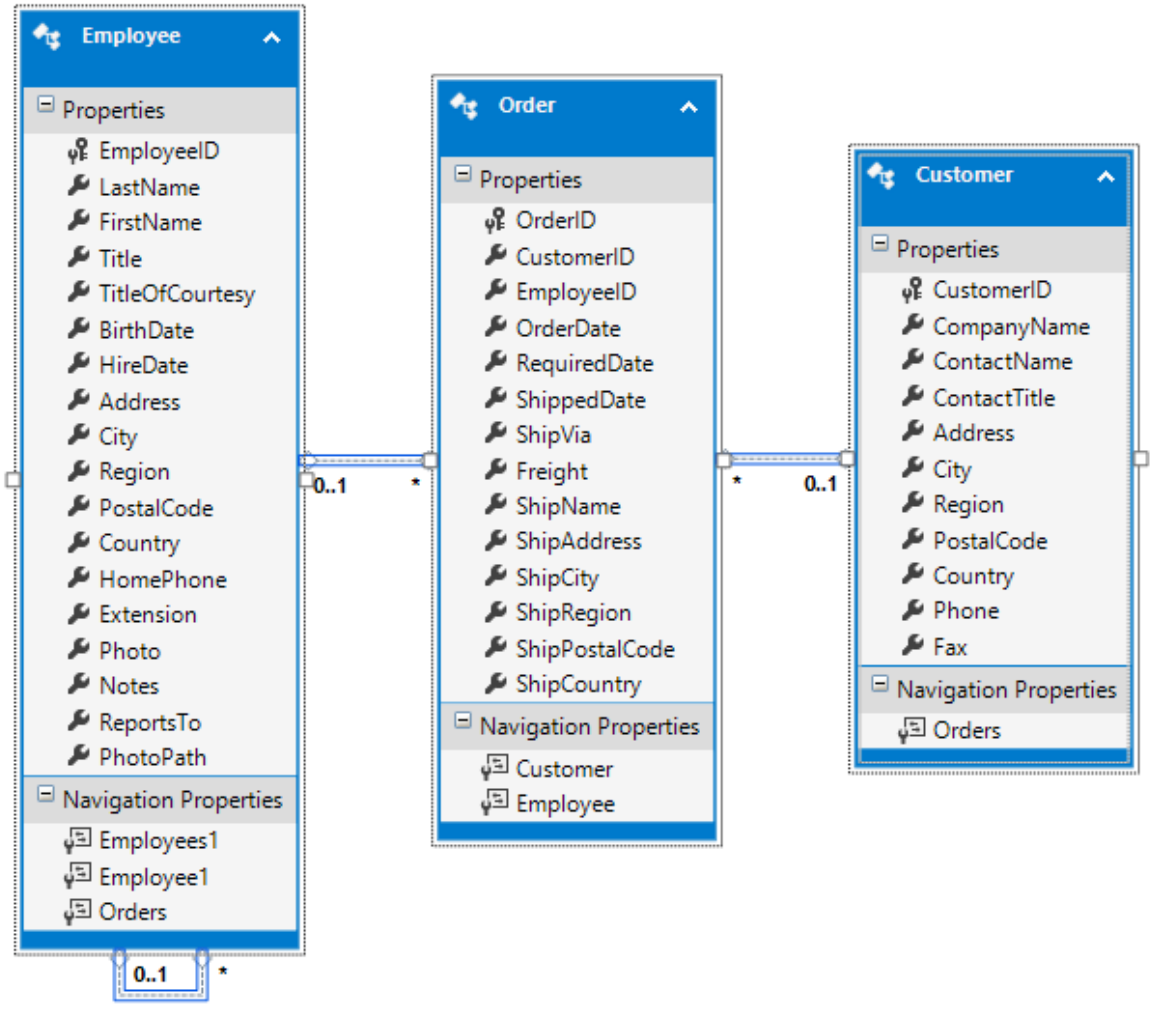


Entity details

Our **SQL tables** have the following **primary keys**:

- **Orders**: **OrderID**
- **Customers**: **CustomerID**
- **Employees**: **EmployeeID**

The **Orders** table has also two **foreign keys**, **CustomerID** and **EmployeeID**, which define two many-to-one relationships **(* – 0..1**), as indicated by the following diagram:



Special attention is required for the **Order.EmployeeID** column, which may be **null** - e.g. for an **Order** which was not yet allocated to a specific **Employee**.

Our **XML** versions of these tables - **XOrders.xml**, **XCustomers.xml** and **XEmployees.xml -** are trimmed, as we have removed quite a few columns which are not required in this assignment.

In the **XML** versions, the relationships between these entities has been lost, but can be **reconstructed** from the existing **primary** and **foreign** keys.

Response data from services

The **OData** services honour all requests which can be answered based on their data (SQL database or XML files).

The **MVC-SQL** service offers a **REST** method called **Sql/WebGrid** which returns an **HTML TABLE** containing a page of the **LEFT JOIN** between table **Orders** and tables **Customers** and **Employees**.

This table contains the following columns:

- **OrderID**: from **Orders OrderID**
- **OrderDate**: from **Orders OrderDate**
- **Freight**: from **Orders Freight**
- **ShipCity**: from **Orders ShipCity**
- **ShipCountry**: from **Orders ShipCountry**
- **CompanyName**: from associated **Customer**.**CompanyName**
- **ContactName**: from associated **Customer**.**ContactName**
- **EmpFirstName**: from associated **Employee**.**FirstName**; or **null**, if there is no associated **Employee**
- **EmpLastName**: from associated **Employee**.**LastName**; or **null**, if there is no associated **Employee**

The last two columns have new names, obtained by prepending **Emp** to the actual column names in the **Employees** table.

Note that an **INNER JOIN** will skip the **Order** rows with no associated **Employee**.

The header display names will contain extra spaces in combined words, e.g. text **Order ID** for column **OrderID**.

The table can be **sorted** by clicking on any of the column headers, as usually alternating between **ascending** and **descending** sorting orders.

To avoid ambiguities, if the primary sorting is on any other column except **OrderID**, then the sorting will also use **OrderId** as a secondary criterion, in the same direction as the primary sort. For example, these are possible sorting criteria:

- **OrderID ASC**
- **OrderID DESC**
- **EmpFirstName ASC, OrderID ASC**
- **EmpFirstName DESC, OrderID DESC**

Important: **server-side sorting and pagination**! For this assignment, all sorting and pagination must be done by the **SQL Server**! Your app must only get the required rows to be displayed in the table!

The **Sql/WebGrid** REST method has four **parameters** which specify: **sortCol** – the primary sorting column, **sortDir** – the sorting direction, **rowsPerPage** – the page size, and **page** - the page number.

Required implementations

Please follow the following indications, which will ensure a fair marking on our COMPSCI lab machines.

All services must be built in **C#**, with **Visual Studio 2015**, using the **Web Application** solution.

- **OData-XML** uses a **Wcf Data Service** template, with a manually written data source, which retrieves data from the **C:\usertmp\ XML** files and rebuilds their associations.
- **OData-SQL** uses a **Wcf Data Service** template, with an automatically generated **ADO.NET EF** data source, based on the **C:\usertmp\Northwind.mdf** database file, accessed via the **SQL 2016 LocalDB** service– i.e. **(LocalDb)\MSSQLLocalDB**.
- **MVC-SQL** uses an **MVC 5** template, with an automatically generated **ADO.NET EF** data source, based on the same database file.

  You will need to add a custom controller called **SqlController**, with one method called **WebGrid**, and its corresponding view **Sql\WebGrid.cshtml**.

  For a compact coding, use **Dynamic LINQ**, as defined in **Dynamic.cs.**

  You also need to insert a new button, **SQL WebGrid**, in the default generated global layout, **_Layout.cshtml**.

We emphasize that you must build your projects entirely in **C#**, with the exception of the MVC-SQL UI, where you can add – if you wish – UI specific CSS or JavaScript code.

Specifically, you must develop high-level functional C# code (as discussed in the lectures), which will be automatically translated to SQL, as needed. No SQL code should be present in any of your files.

Suggestion to start by making yourself familiar with the three SQL tables – best using **Linqpad** - and pay special attention to all **nullable** fields, such as Order.EmployeeID.

The marking will be done semi-automatically, using tools. There will be quite a few test cases and each pass/fail will be decided based on exact text comparisons via **diff**. We will publish a sample of test scenarios, with expected outputs.

Please talk to us if you wish to experiment with other possible implementations, e.g. Web API, F#, WebSharper… Our general advice is to first solve the problems exactly as indicated above, and only then try other approaches – small bonuses might be possible, but these need to be agreed before.

Querying the OData-XML service – sample fragments via RESTClient

Assuming that the service has default name **WcfDataService1** and is available at port **8181** on the local machine

- Request: http://localhost:8181/WcfDataService1.svc/

  Response:

```
<?xml version="1.0" encoding="utf-8"?>
<service xml:base="http://localhost:8181/WcfDataService1.svc/" x
/2005/Atom">
  <workspace>
    <atom:title>Default</atom:title>
    <collection href="Customers">
      <atom:title>Customers</atom:title>
    </collection>
    <collection href="Employees">
      <atom:title>Employees</atom:title>
    </collection>
    <collection href="Orders">
      <atom:title>Orders</atom:title>
    </collection>
  </workspace>
</service>
```

- Request: http://localhost:8181/WcfDataService1.svc/Orders()?$format=json

  Response:

```
{
    "odata.metadata": "http://localhost:8181/WcfDataService1.svc/$metadata#Orders",
    "value":
    [
        {
            "OrderID": 10248,
            "CustomerID": "VINET",
            "EmployeeID": 5,
            "OrderDate": "1996-07-04T00:00:00",
            "ShippedDate": "1996-07-16T00:00:00",
            "Freight": "32.3800",
            "ShipName": "Vins et alcools Chevalier",
            "ShipCity": "Reims",
            "ShipCountry": "France"
        },
        {
```

- Request: http://localhost:8181/WcfDataService1.svc/Customers()?$format=json

  Response:


```
{
    "odata.metadata": "http://localhost:8181/WcfDataService1.svc/$metadata#Customers",
    "value":
    [
        {
            "CustomerID": "ALFKI",
            "CompanyName": "Alfreds Futterkiste",
            "ContactName": "Maria Anders"
        },
        {
            "CustomerID": "ANATR",
            "CompanyName": "Ana Trujillo Emparedados y helados",
            "ContactName": "Ana Trujillo"
        },
        {
            "CustomerID": "ANTON",
            "CompanyName": "Antonio Moreno Taquería",
            "ContactName": "Antonio Moreno"
        },
        {
```


- Request: http://localhost:8181/WcfDataService1.svc/Employees()?$format=json

  Response:


```
{
    "odata.metadata": "http://localhost:8181/WcfDataService1.svc/$metadata#Employees",
    "value":
    [
        {
            "EmployeeID": 1,
            "FirstName": "Nancy",
            "LastName": "Davolio"
        },
        {
            "EmployeeID": 2,
            "FirstName": "Andrew",
            "LastName": "Fuller"
        },
        {
            "EmployeeID": 3,
            "FirstName": "Janet",
            "LastName": "Leverling"
        },
        {
```

- Request:
  http://localhost:8181/WcfDataService1.svc/Orders()?$orderby=OrderID&$expand=Customer,Employee&$format=json

  Response:


```
{
    "odata.metadata": "http://localhost:8181/WcfDataService1.svc/$metadata#Orders",
    "value":
    [
        {
            "Customer":
            {
                "CustomerID": "VINET",
                "CompanyName": "Vins et alcools Chevalier",
                "ContactName": "Paul Henriot *******"
            },
            "Employee":
            {
                "EmployeeID": 5,
                "FirstName": "Steve",
                "LastName": "Buchanan"
            },
            "OrderID": 10248,
            "CustomerID": "VINET",
            "EmployeeID": 5,
            "OrderDate": "1996-07-04T00:00:00",
            "ShippedDate": "1996-07-16T00:00:00",
            "Freight": "32.3800",
            "ShipName": "Vins et alcools Chevalier",
            "ShipCity": "Reims",
            "ShipCountry": "France"
        },
        {
```

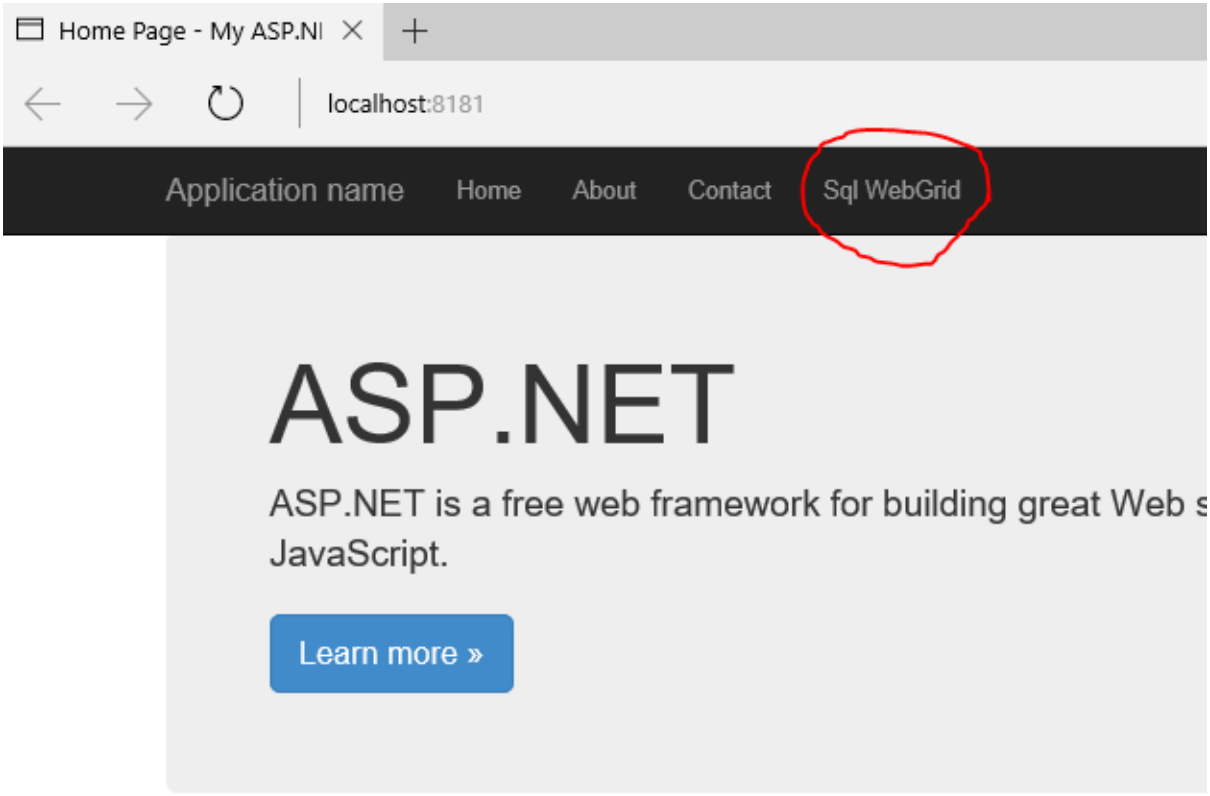

OData-SQL queries are similar

But will return more data, as the database tables have more columns than the trimmed XML documents.

Querying the MVC-SQL service - sample fragments via browser

Assuming that the service is available at port **8181** on the local machine

- The base request: http://localhost:8181/
  Note your additional button, **Sql WebGrid**, on the default page (which was constructed by the VS MVC 5 support).



- General format – here shown with all default parameter values

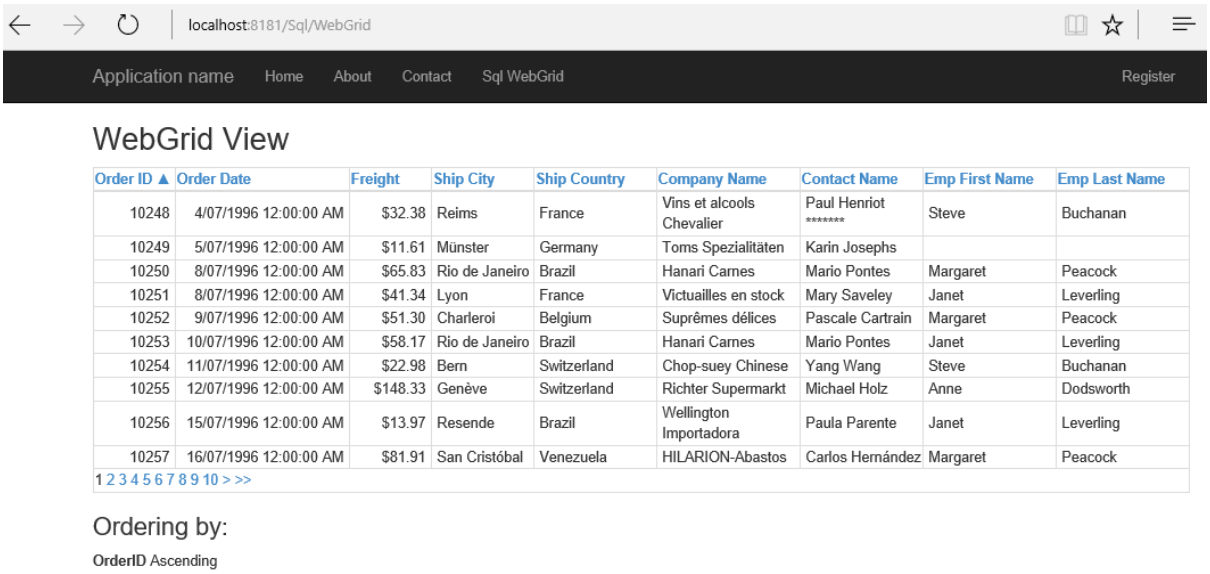
http://localhost:8181/Sql/WebGrid?page=1&rowsPerPage=10&sortCol=OrderID&sortDir=ASC

Thus, query parameters are:

  - **page**: int, starting with 1
  - **rowsPerPage**: int
  - **sortCol**: string
  - **sortDir**: string, ASC or DESC

- Request: http://localhost:8181/Sql/WebGrid

  This request takes all default parameter values – page: 1, …



Notes (not all these features are exemplified here):

- table with borders
- header of primary sorted column has up/down arrows (here OrderID ▲)
- primary sorting possible on all columns, coming from all tables, Orders, Customers, Employees
- secondary sorting on OrderID, in the same direction as the primary sorting (not relevant here)
- freight has currency formatting
- numbers are right aligned
- table footer has a pager with usual options (first, previous, page numbers, next, last)
- Ordering by indicates the current primary sorting column and direction (here OrderID ASC)
- The automatically generated SQL code is displayed in a following textarea box (next page)

Request SQL:

```
      SELECT
   [Project1].[OrderID] AS [OrderID],
   [Project1].[OrderDate] AS [OrderDate],
   [Project1].[Freight] AS [Freight],
   [Project1].[ShipCity] AS [ShipCity],
   [Project1].[ShipCountry] AS [ShipCountry],
   [Project1].[CompanyName] AS [CompanyName],
   [Project1].[ContactName] AS [ContactName],
   [Project1].[C1] AS [C1],
   [Project1].[C2] AS [C2]
   FROM ( SELECT
      [Extent1].[OrderID] AS [OrderID],
      [Extent1].[OrderDate] AS [OrderDate],
      [Extent1].[Freight] AS [Freight],
      [Extent1].[ShipCity] AS [ShipCity],
      [Extent1].[ShipCountry] AS [ShipCountry],
      [Extent2].[CompanyName] AS [CompanyName],
      [Extent2].[ContactName] AS [ContactName],
      CASE WHEN ([Extent3].[EmployeeID] IS NULL) THEN CAST(NULL AS varchar(1)) ELSE
[Extent3].[FirstName] END AS [C1],
      CASE WHEN ([Extent3].[EmployeeID] IS NULL) THEN CAST(NULL AS varchar(1)) ELSE
[Extent3].[LastName] END AS [C2]
      FROM   [dbo].[Orders] AS [Extent1]
      LEFT OUTER JOIN [dbo].[Customers] AS [Extent2] ON [Extent1].[CustomerID] = [Extent2].
[CustomerID]
      LEFT OUTER JOIN [dbo].[Employees] AS [Extent3] ON [Extent1].[EmployeeID] = [Extent3].
[EmployeeID]
   )  AS [Project1]
   ORDER BY [Project1].[OrderID] ASC
   OFFSET 0 ROWS FETCH NEXT 10 ROWS ONLY
```

- Request: http://localhost:8181/Sql/WebGrid?page=20&rowsPerPage=5&sortCol=EmpFirstName&sortDir=DESC

WebGrid View

Order ID	Order Date	Freight	Ship City	Ship Country	Company Name	Contact Name	Emp First Name ▼	Emp Last Name
10497	4/04/1997 12:00:00 AM	$36.21	Frankfurt a.M.	Germany	Lehmanns Marktstand	Renate Messner	Robert	King
10496	4/04/1997 12:00:00 AM	$46.77	Sao Paulo	Brazil	Tradição Hipermercados	Anabela Domingues	Robert	King
10490	31/03/1997 12:00:00 AM	$210.19	San Cristóbal	Venezuela	HILARION-Abastos	Carlos Hernández	Robert	King
10483	24/03/1997 12:00:00 AM	$15.28	Seattle	USA	White Clover Markets	Karl Jablonski	Robert	King
10458	26/02/1997 12:00:00 AM	$147.06	Charleroi	Belgium	Suprêmes délices	Pascale Cartrain	Robert	King

<< < 16 17 18 19 20 21 22 23 24 25 > >>

Ordering by:

EmpFirstName Descending

Request SQL:

```
    SELECT
  [Project1].[OrderID] AS [OrderID],
  [Project1].[OrderDate] AS [OrderDate],
  [Project1].[Freight] AS [Freight],
  [Project1].[ShipCity] AS [ShipCity],
  [Project1].[ShipCountry] AS [ShipCountry],
  [Project1].[CompanyName] AS [CompanyName],
  [Project1].[ContactName] AS [ContactName],
  [Project1].[C1] AS [C1],
  [Project1].[C2] AS [C2]
  FROM ( SELECT
     [Extent1].[OrderID] AS [OrderID],
     [Extent1].[OrderDate] AS [OrderDate],
     [Extent1].[Freight] AS [Freight],
     [Extent1].[ShipCity] AS [ShipCity],
     [Extent1].[ShipCountry] AS [ShipCountry],
     [Extent2].[FirstName] AS [FirstName],
     [Extent3].[CompanyName] AS [CompanyName],
     [Extent3].[ContactName] AS [ContactName],
     CASE WHEN ([Extent2].[EmployeeID] IS NULL) THEN CAST(NULL AS varchar(1)) ELSE
[Extent2].[FirstName] END AS [C1],
     CASE WHEN ([Extent2].[EmployeeID] IS NULL) THEN CAST(NULL AS varchar(1)) ELSE
[Extent2].[LastName] END AS [C2]
     FROM  [dbo].[Orders] AS [Extent1]
     LEFT OUTER JOIN [dbo].[Employees] AS [Extent2] ON [Extent1].[EmployeeID] = [Extent2].
[EmployeeID]
     LEFT OUTER JOIN [dbo].[Customers] AS [Extent3] ON [Extent1].[CustomerID] = [Extent3].
[CustomerID]
  ) AS [Project1]
  ORDER BY [Project1].[FirstName] DESC, [Project1].[OrderID] DESC
  OFFSET 95 ROWS FETCH NEXT 5 ROWS ONLY
```

Additional readings

OData

- WCF Data Services
  https://msdn.microsoft.com/en-us/library/cc668792(v=vs.103).aspx
- Exposing Your Data as an OData Service (WCF Data Services)
  https://msdn.microsoft.com/en-us/library/dd728286(v=vs.103).aspx
- Creating the Northwind Data Service (WCF Data Services Quickstart)
  https://msdn.microsoft.com/en-us/library/dd728275(v=vs.103).aspx
- Configuring the Data Service (WCF Data Services)
  https://msdn.microsoft.com/en-us/library/ee358710(v=vs.103).aspx
- How to: Create a Data Service Using the Reflection Provider (WCF Data Services)
  https://msdn.microsoft.com/en-us/library/dd728281(v=vs.103).aspx
- Accessing an OData Service (WCF Data Services)
  https://msdn.microsoft.com/en-us/library/dd728283(v=vs.103).aspx
- Accessing OData Feeds from a Web Browser (WCF Data Services Quickstart)
  https://msdn.microsoft.com/en-us/library/dd728279(v=vs.103).aspx

MVC

- Getting Started with ASP.NET MVC 5
  http://www.asp.net/mvc/overview/getting-started/introduction/getting-started
  Follow this tut, not the Core MVC 6 version
- GridView Class
  https://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.gridview(v=vs.110).aspx

Assignment marks

This assignment is worth 10/100 course marks and will be marked out of 100 (so each assignment mark will translate into 0.1 course marks). The three projects will be marked as indicated below:

- **mvc-sql**: 55 marks
- **odata-xml**: 30 marks
- **odata-sql**: 15 marks

There are two possible small bonuses:

- **odata-xml**: 5 marks, if successfully published and accessible on the cloud (Azure)
- **odata-sql**: 5 marks, if successfully published and accessible on the cloud (Azure)

Other small bonuses may be awarded for outstanding projects. Please check with us before attempting something special.

Deliverables and Submission

Submit electronically, to the COMPSCI web dropbox, an **upi.7z** archive containing three folders, one for each of the three VS solutions – these three folders should be correspondingly named: **mvc-sql**, **odata-sql**, **odata-xml**.

The submission size will be **limited to 5 MB** per student - for all three solutions together. Therefore, before archiving these, please **clean** all your projects and **delete all their packages**. The markers will restore all needed packages and rebuild your projects.

Please recheck your projects before submitting, starting from the archive prepared for submission (which is the only thing that the marker will receive from you). Expand this archive on a lab machine, in another folder, then restore the required packages, rebuild your solutions and test them again!

Also, please do **not** submit any **SQL** database or **XML** files. The markers will already have their own SQL database and XML files in their own C:\usertmp\ folders.

Pay special attention while developing the SQL projects, as VS will try to "lure" you into accepting a copy of the **SQL** database file into your project's **App_Data** folder – say **NO** to this attempt!

Please note the strict COMPSCI policy on **plagiarism**. In particular, build your own VS solutions from scratch, starting with **new solutions** and **new projects**. Developing your own project on a project previously created by someone else will be considered a plagiarism attempt.

Please keep your electronic dropbox receipt!

Deadline

Monday 12 Sept 2016, 18:00!

Please do not leave it for the last minute. Remember that you can resubmit and, by default, we only consider your last submission.

After this deadline, submissions will be still accepted for 4 more days, with gradually increasing penalties, of 0.5% for each hour late.

For example:

Monday 12 Sept 2016, 20:00:	-1%
Tuesday 13 Sept 2016, 18:00:	-12%
Tuesday 13 Sept 2016, 20:00:	-13%
Wednesday 14 Sept 2016, 18:00:	-24%
Thursday 15 Sept 2016, 18:00:	-36%
Friday 16 Sept 2016, 18:00:	-48%

After this, no more submissions are accepted!