



locane

Interview Practical Report Analyst

Author: Tom Aylen
Version: 1
Release Date: 02/08/2021
Status: Release

Better for business

Contents

1	Requirements	3
1.1	Software	3
1.2	Supplied Files	4
2	Instructions	4
2.1	Install Northwind Database	4
2.2	SSRS Practical	4
2.2.1	Add Product Total	5
2.2.2	Add Order Total	6
2.2.3	Add Date Parameters	6
2.2.4	Create Column Chart	7
2.2.5	SQL Query	7
2.3	VB Practical	9
2.3.1	Find and Fix Fault	10
2.3.2	Coding	10
3	Completion	12
3.1	Provide Files to locane	12

1 Requirements

1.1 Software

The following software is required for to complete the practical interview.

Visual Studio Community

<https://visualstudio.microsoft.com/downloads/>

SQL Server Data Tools (SSDT) for Visual Studio

<https://docs.microsoft.com/en-us/sql/ssdt/download-sql-server-data-tools-ssdt?view=sql-server-ver15>

Microsoft Reporting Services Projects Extension for Visual Studio

Installed from Visual Studio menu: Extensions > search for name

<https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftReportProjectsforVisualStudio>

SQL Server Express

The Visual Studio Project is configured to use the default server name of './SQLEXPRESS'. Using the same name will avoid having to change it in the project.

<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

SQL Server Management Studio

<https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

1.2 Supplied Files

The following listed files are supplied, these can be downloaded via GitHub at the below link.

The files constitute a Visual Studio Solution which contains 2 Projects.

<https://github.com/locaneDevApps/report-analyst-practical>

File Type	Name
Solution File	locane Report Analysis Practical.sln
Visual Studio Project Folder <i>This is a SSRS Report Project</i>	SSRS Practical
Project File	SSRS Practical.rptproj
SSRS Report	Activity1.rdl
Visual Studio Project Folder <i>This is a .NET Core Console App</i>	VB Practical
Folder	bin
Folder	obj
Visual Basic Source Code	Activity2.vb

2 Instructions

After downloading and installing the required software, the Northwind database will need to be installed to the local SQL Instance.

2.1 Install Northwind Database

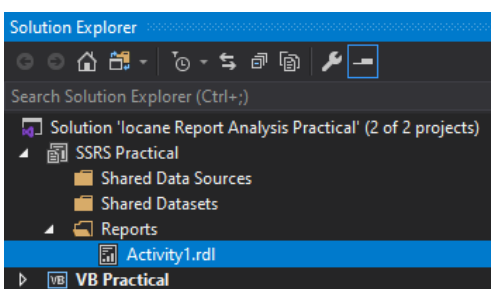
Open SSMS (SQL Server Management Studio), connect to the local SQL instance on your machine, open a new query. Within the query copy the code from the script instnwnd.sql and run it.

<https://github.com/Microsoft/sql-server-samples/tree/master/samples/databases/northwind-pubs>

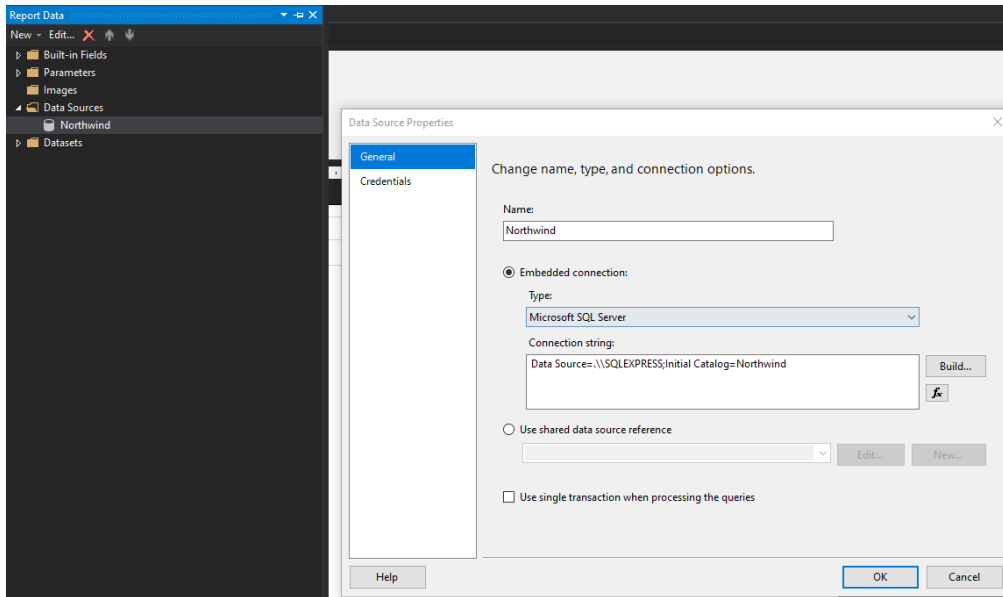
2.2 SSRS Practical

The SSRS (SQL Server Reporting Services) portion of the test will use the Northwind database. Ensure the server name is correct. If the default SQL Express instance is not used it will need to be updated.

Open the solution file in Visual Studio. Open the SSRS Project, and the Activity1.rdl file.



Ensure SQL database server name and instance is correct in the Northwind Data Source.



2.2.1 Add Product Total

The "Activity1.rdl" file has a Dataset called "Orders" and table already created. Add a 'Total' column on the right hand side of the table. The value in the "Total" column should be calculated based on the unit price, quantity and discount of the product in the order. Please refer to the below screenshot for how this should look.

Example

Select Customer: Alfreds Futterkiste						
Order ID	Order Date	Product Name	Unit Price	Quantity	Discount	Total
10643	25/08/1997	Rössle Sauerkraut	\$45.60	15	0.25	\$513.00
10643	25/08/1997	Chartreuse verte	\$18.00	21	0.25	\$283.50
10643	25/08/1997	Spegesild	\$12.00	2	0.25	\$18.00
10692	03/10/1997	Vegie-spread	\$43.90	20	0	\$878.00
10702	13/10/1997	Aniseed Syrup	\$10.00	6	0	\$60.00
10702	13/10/1997	Lakkalikööri	\$18.00	15	0	\$270.00
10835	15/01/1998	Raclette Courdavault	\$55.00	15	0	\$825.00
10835	15/01/1998	Original Frankfurter grüne Soße	\$13.00	2	0.2	\$20.80
10952	16/03/1998	Grandma's Boysenberry Spread	\$25.00	16	0.05	\$380.00
10952	16/03/1998	Rössle Sauerkraut	\$45.60	2	0	\$91.20
11011	09/04/1998	Escargots de Bourgogne	\$13.25	40	0.05	\$503.50
11011	09/04/1998	Flotemysost	\$21.50	20	0	\$430.00

2.2.2 Add Order Total

Using the existing table, create an 'Order Total' that displays the 'Total' cost for each Order. The final result should look like the below.

Example

Select Customer: Alfreds Futterkiste						
Find Next						
Order ID	Order Date	Product Name	Unit Price	Quantity	Discount	Total
10643	25/08/1997	Rössle Sauerkraut	\$45.60	15	0.25	\$513.00
	25/08/1997	Chartreuse verte	\$18.00	21	0.25	\$283.50
	25/08/1997	Spegesild	\$12.00	2	0.25	\$18.00
					Order Total	\$814.50
10692	03/10/1997	Vegie-spread	\$43.90	20	0.00	\$878.00
					Order Total	\$878.00
10702	13/10/1997	Aniseed Syrup	\$10.00	6	0.00	\$60.00
	13/10/1997	Lakkalikööri	\$18.00	15	0.00	\$270.00
					Order Total	\$330.00
10835	15/01/1998	Raclette Courdavault	\$55.00	15	0.00	\$825.00
	15/01/1998	Original Frankfurter grüne Soße	\$13.00	2	0.20	\$20.80
					Order Total	\$845.80
10952	16/03/1998	Grandma's Boysenberry Spread	\$25.00	16	0.05	\$380.00
	16/03/1998	Rössle Sauerkraut	\$45.60	2	0.00	\$91.20
					Order Total	\$471.20
11011	09/04/1998	Escargots de Bourgogne	\$13.25	40	0.05	\$503.50
	09/04/1998	Flotemysost	\$21.50	20	0.00	\$430.00
					Order Total	\$933.50

2.2.3 Add Date Parameters

Add two new parameters that are used to filter the Orders by the Order Date. Orders should appear that have an Order Date between the new Start Date parameter and the new End Date parameter.

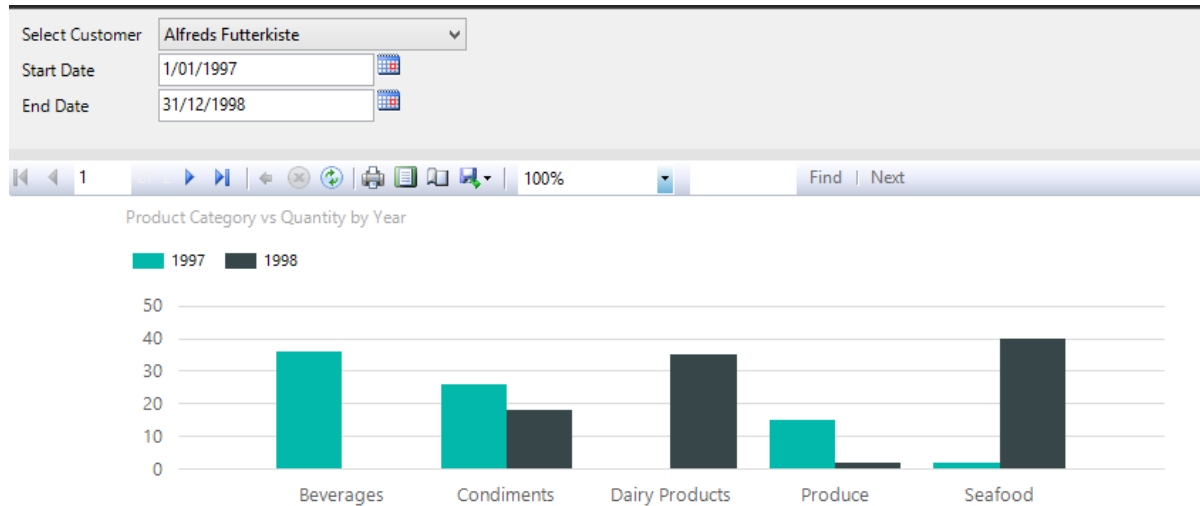
Example

Select Customer: Alfreds Futterkiste						
Start Date: 1/01/1998						
End Date: 30/06/1998						
Find Next						
Order ID	Order Date	Product Name	Unit Price	Quantity	Discount	Total
10835	15/01/1998	Raclette Courdavault	\$55.00	15	0.00	\$825.00
	15/01/1998	Original Frankfurter grüne Soße	\$13.00	2	0.20	\$20.80
					Order Total	\$845.80
10952	16/03/1998	Grandma's Boysenberry Spread	\$25.00	16	0.05	\$380.00
	16/03/1998	Rössle Sauerkraut	\$45.60	2	0.00	\$91.20
					Order Total	\$471.20
11011	09/04/1998	Escargots de Bourgogne	\$13.25	40	0.05	\$503.50
	09/04/1998	Flotemysost	\$21.50	20	0.00	\$430.00
					Order Total	\$933.50
Grand Total						\$2250.50

2.2.4 Create Column Chart

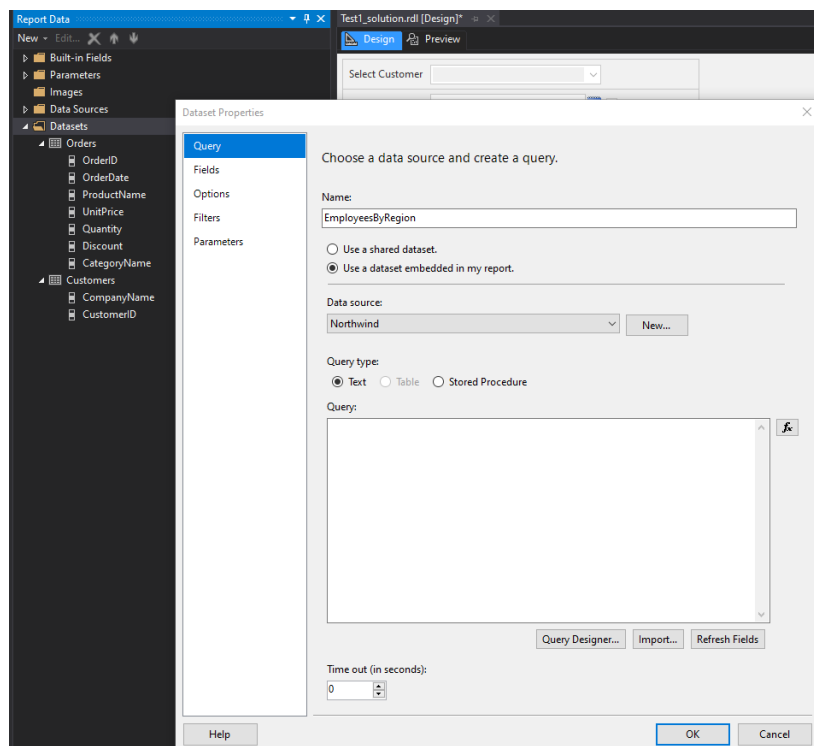
Using the existing "Orders" dataset, create a column chart showing the total quantity of products for each category for each year. The resulting chart should look like the below.

Example



2.2.5 SQL Query

Create a new embedded Dataset as shown below. The dataset needs to have a SQL query that will find the number of Territories in each Region, as well as the number of unique employees that work in each region. All totals must be calculated using SQL. Create a simple table in "Activity1.rdl" to display the query results.

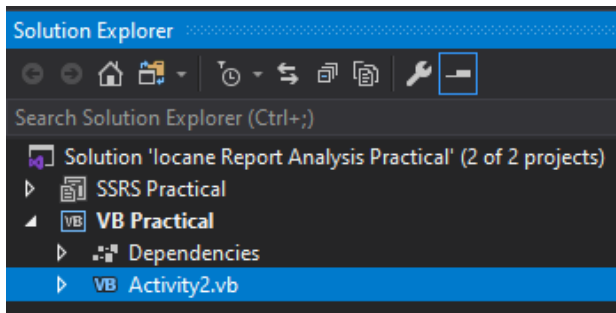


Example Results

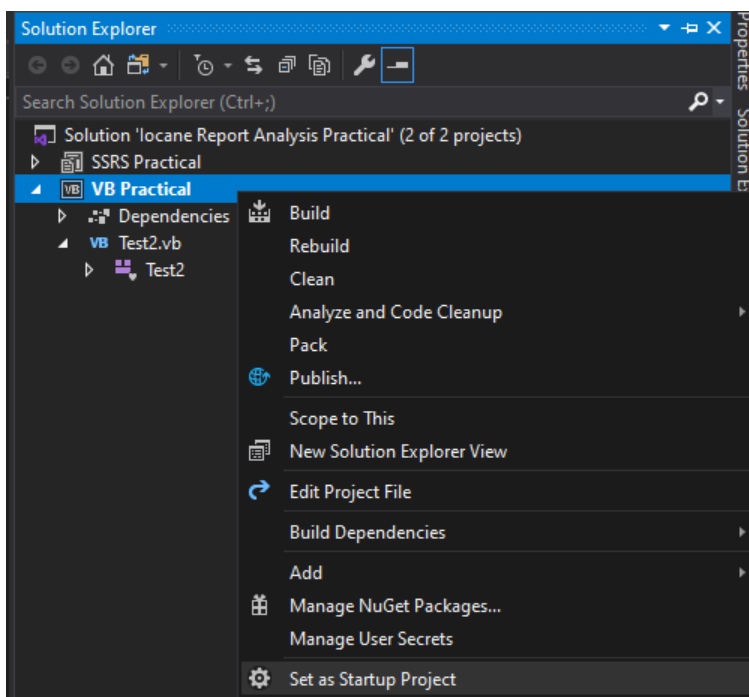
Region ID	Region Description	count territories	count employee
1	Eastern	19	4
2	Western	15	2
3	Northern	11	2
4	Southern	4	1

2.3 VB Practical

The Visual Basic practical uses a Console App project. Open the VB Source file Activity2.vb



Set the project as the Startup Project by right-clicking the project and selecting 'Set as Startup Project'

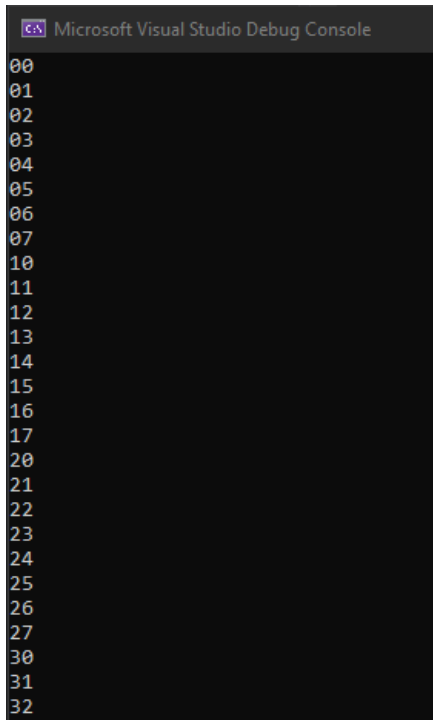


This allows the app to be tested by clicking the  button.

2.3.1 Find and Fix Fault

There is a subroutine in Activity2.vb called "FindAndFixFault()" that should print out a sequence of numbers in octal (base 8). The code currently does not do this, it prints another sequence. Find and fix the problem.

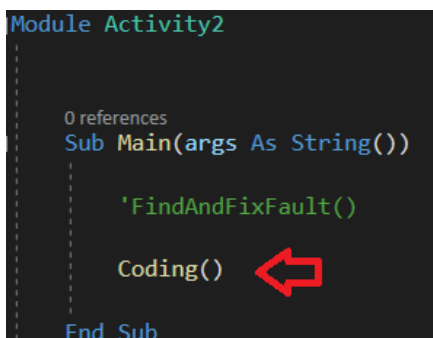
Example



```
Microsoft Visual Studio Debug Console
00
01
02
03
04
05
06
07
10
11
12
13
14
15
16
17
20
21
22
23
24
25
26
27
30
31
32
```

2.3.2 Coding

There is a subroutine in Activity2.vb called "Coding()" where you will be asked to create some code. First of all uncomment the "Coding()" call in the "Main" method as shown below. Optionally the "FindAndFixFault()" can be commented out.



```
Module Activity2
    0 references
    Sub Main(args As String())
        'FindAndFixFault()
        Coding()
    End Sub
```

The "Coding()" subroutine has some pre-existing code which creates a dictionary of cells, populates the cell values and writes the dictionary to the console. Cells A:1 to A:10 have order numbers, cells B:1 to B:10 have dates, cells C:1 to C:10 have monetary values.

```
Sub Coding()

    Dim cells As New Dictionary(Of String, Object)()

    For i As Integer = 1 To 10
        cells(String.Format("A:{0}", i)) = String.Format("Order #{0}", i)
        cells(String.Format("B:{0}", i)) = DateTime.Now.ToString
        cells(String.Format("C:{0}", i)) = i * 18
    Next

    For Each pair As KeyValuePair(Of String, Object) In cells
        Console.WriteLine(pair)
    Next

End Sub
```

2.3.2.1 Cell Dates

Modify the date values of cells "B:1" through "B:9", set each date to a random date prior to today. Also format ALL dates in the dictionary. For example 04/08/2021 should appear as "4 Aug 21".

2.3.2.2 Cell Currency

Modify the currency values of the following cells by multiplying the existing value by the value specified below.

"C:2" x 20, "C:4" x 40, "C:6" x 60, "C:8" x 80, "C:10" x 100

Format the cells "C:1" through "C:10" as currency, example format "\$120.00".

2.3.2.3 Switch Cells

Switch the values of cells "C:2" and "C:7".

2.3.2.4 Print Table

Print the cells in a table format to the console as shown below.




Order #1	29 Aug 19	\$18.00
Order #2	17 Feb 20	\$126.00
Order #3	2 Jan 20	\$54.00
Order #4	18 Oct 20	\$2,880.00
Order #5	6 Oct 20	\$90.00
Order #6	21 Jun 19	\$6,480.00
Order #7	20 Jul 21	\$720.00
Order #8	5 Jul 19	\$11,520.00
Order #9	12 May 19	\$162.00
Order #10	5 Aug 21	\$18,000.00

3 Completion

3.1 Provide Files to iocane

Upon completion, please email the Solution folder in a single zipped file to DevApps@iocane.com.au.

The files and folders shown below should be included in the zip file.

Name	Date modified	Type
 SSRS Practical	4/08/2021 12:52 PM	File folder
 VB Practical	4/08/2021 12:49 PM	File folder
 iocane Report Analysis Practical.sln	4/08/2021 12:32 PM	Microsoft Visual Studio Solution