SSRS –Integration-Microsoft Dynamics CRM

SQL vs. FetchXML reports in CRM

When it comes to building reports for CRM, there are two options available: SQL reports or FetchXML reports. Both make use of SQL Reporting Services and require the SSRS Data Connector to be installed to interface with CRM. In the following table I am attempting to summarize the differences between these two reporting options:

	SQL Reports	**FetxhXML Reports**
Building experience	Requires a separate program for designing the report such as SQL Business Intelligence Development Studio (BIDS) or Report Builder.	CRM comes with a Report Wizard which can be used for building these reports. The report wizard is a web report designer integrated with CRM. These reports can also be designed using BIDS (must download the Report Authoring Extension).
Skill set	Building SQL reports requires SQL Server skills and development experience.	When built using the Report Wizard, advanced CRM users can have the skills to build reports (or super-users) without requiring a developer.
Flexibility	These reports can take data from CRM and present it in multiple ways. Reports can achieve complex requirements as you can use any feature from SQL Reporting Services.	Functionality is restricted to what the Report Wizard can support which can be quite limiting at times.
Queries	Data is queried using SQL statements that read the filtered views in the organization database.	FetchXML queries are used for retrieving data for these reports (Advanced Find can be used to generate FetchXML queries).

Reporting mechanism	These reports can be scheduled, delivered by email and other mechanisms.	Must be executed on-demand.
CRM Online Support	Not supported	Supported

Creating Dynamics CRM Reports in SQL Reporting Services with Pre-Filtering and Default Filters

With the release of Microsoft Dynamics CRM 2011 and the rapid proliferation of through cloud deployments, Microsoft had to make some changes to how SQL Reporting Services worked to make it possible to write SRS Reports without physical database access that could be easily published to either On Premise or Through Cloud deployments of Dynamics CRM 2011. They accomplished that with the release of the Dynamics CRM 2011 Report Authoring Extension--a plug in to SQL Business Development Studio that allows you to select FethXML as a data source when authoring a report in SQL Server Reporting Services.

In this blog post, I am going to walk you through creating an SRS Report and publishing that report to Dynamics CRM 2011. I am also going to show you how to enable the report to use Pre-Filtering and Default Filters. What you will see here will work with On Premise and Through Cloud deployments of Dynamics CRM 2011. In my environment, I will be connecting to a Dynamics CRM 2011 Organization in the NjevityToGo CRM Cloud.

Before we begin, lets talk a little about Pre-Filtering and Default Filters so you have a good understanding of why this is important to your report design.

Pre-Filtering

Have you ever run a report in Dynamics CRM 2011 and been prompted to select whether you want the report to use All Applicable Records, The Selected Records, or All Records on all pages in the current view? Or have you opened the Account Form for a specific account and run a report from the Account Ribbon that only displays data for the account you had opened? In both of these cases, you are using Pre-Filtering.

Pre-Filtering gives you the ability to supply one or more selected records as parameters to the report. This allows the user to select one or more records within the Dynamics CRM 2011 Application and run a report just for those records. It is a much more intuitive way to

supply a parameter to the report than using visible paramters on the report. This should be a basic requirement in most of the reports you author for Dynamics CRM 2011.

Default Filters

When you run a report in Dynamics CRM 2011 that uses Default Filters, there will be an Edit Filters button in the top left corner of the report. Additionally, if you double click a report in the Report List that uses Default Filters, the Default Filters window will open before you are able to run the report. The Report Filtering Criteria window allows you to apply Advanced Find filter criteria to the report using all of fields and relationships on the entities you are reporting against...regardless as to whether or not those fields and relationships exist in the report.

This is a fantastic way for users to supply parameters to reports. It uses the Advanced Find Query Structures, so it is familiar to the users. And it does not require the report designer to know about every possible filter the user might want to use so they can be part of the report. The user can provide default filters and save them into the report so they are used everytime the report runs. And the user can change the filters at any time. The user can even add new filters that were never part of the original design.

I just wrote a report where the requirement was for the report to always show 'This Month's' data. But sometimes they want to be able to change it to show 'last month's' data. Now I could have put a FROM and TO date parameter into the report and created some functions to default them to the beginning and end of the current month. But that is quite a bit of work. And tomorrow, the requirement could have changed to use the Modified Date instead of the Created On date as the basis of the filter. Or another filter could have been added. In the old way of working with parameters, any of these changes would have required additional report design work. But with using Default Filters, I was able to write the report without any parameters, and then setup the default filters using standard Dynamics CRM 2011 capabillities. And the user can change this at any time on their own.

In the following example, I will show you how to do this. So, lets get started.

Authoring a Dynamics CRM 2011 Report that uses Pre-Filtering and Default Filters

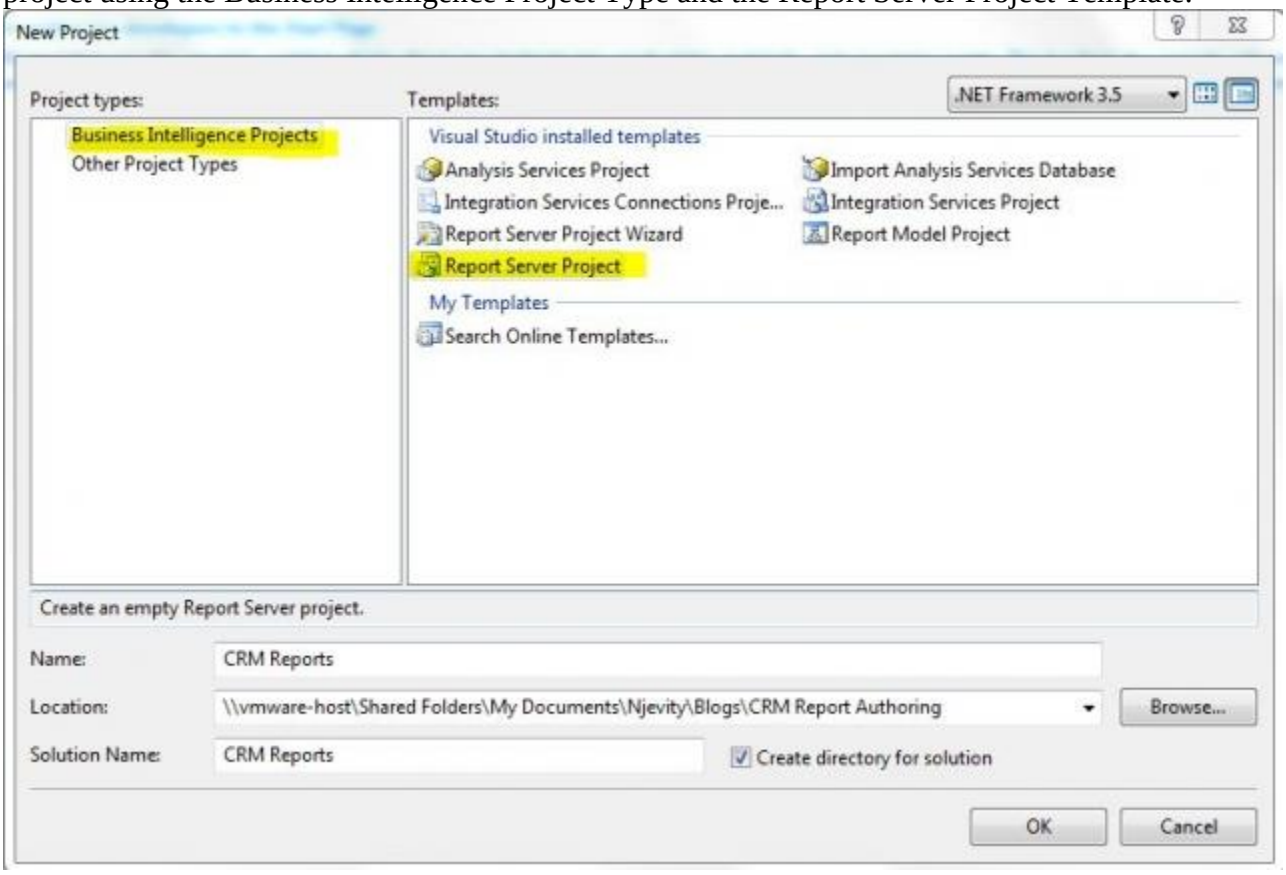
Before we being, you will need the following to author and publish a report for Dynamics CRM 2011:

1. You will need access to a Dynamics CRM 2011 Organization and have the proper security roles to publish reports. This includes:
    1. Organization URL
    2. Organization Unique Name (**Settings >> Customization >> Developer Resources**)

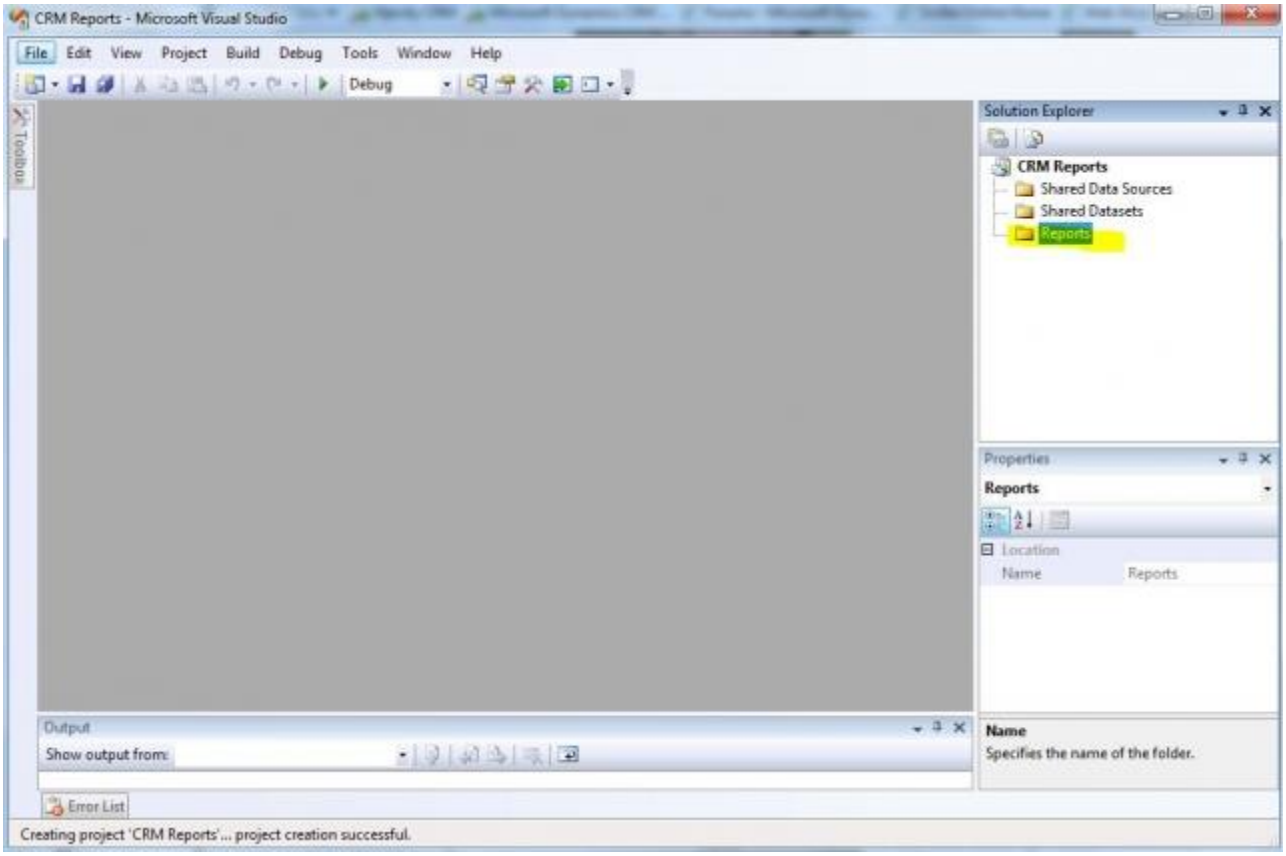
3. UserName and Password for that organization

2. You will need to have SQL Server Business Intelligence Development Studio Installed with the SQL Reporting Services components. (I am using version 2008 R2)
3. You will need to download and install the Dynamics CRM 2011 Report Authoring Extension. You can download it here.

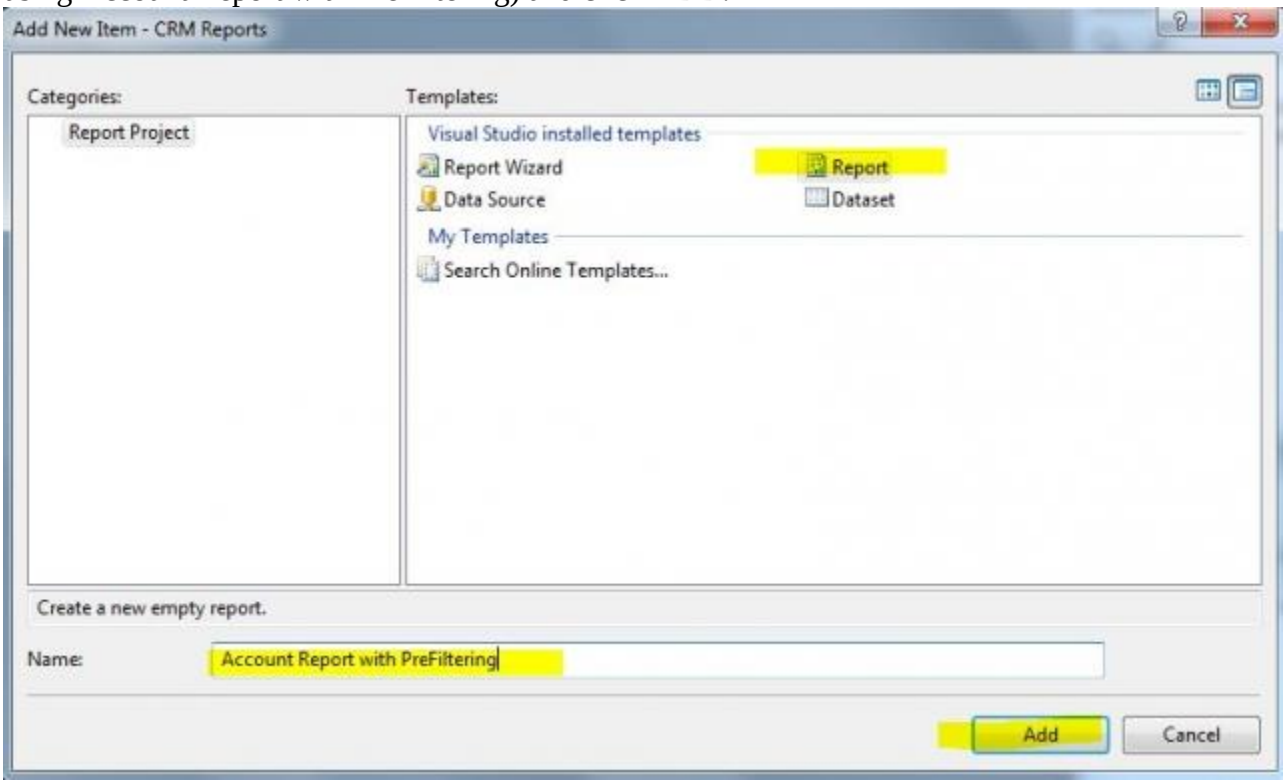
First, we will need to open the Business Intelligence Development Studio and create a new project using the Business Intelligence Project Type and the Report Server Project Template.



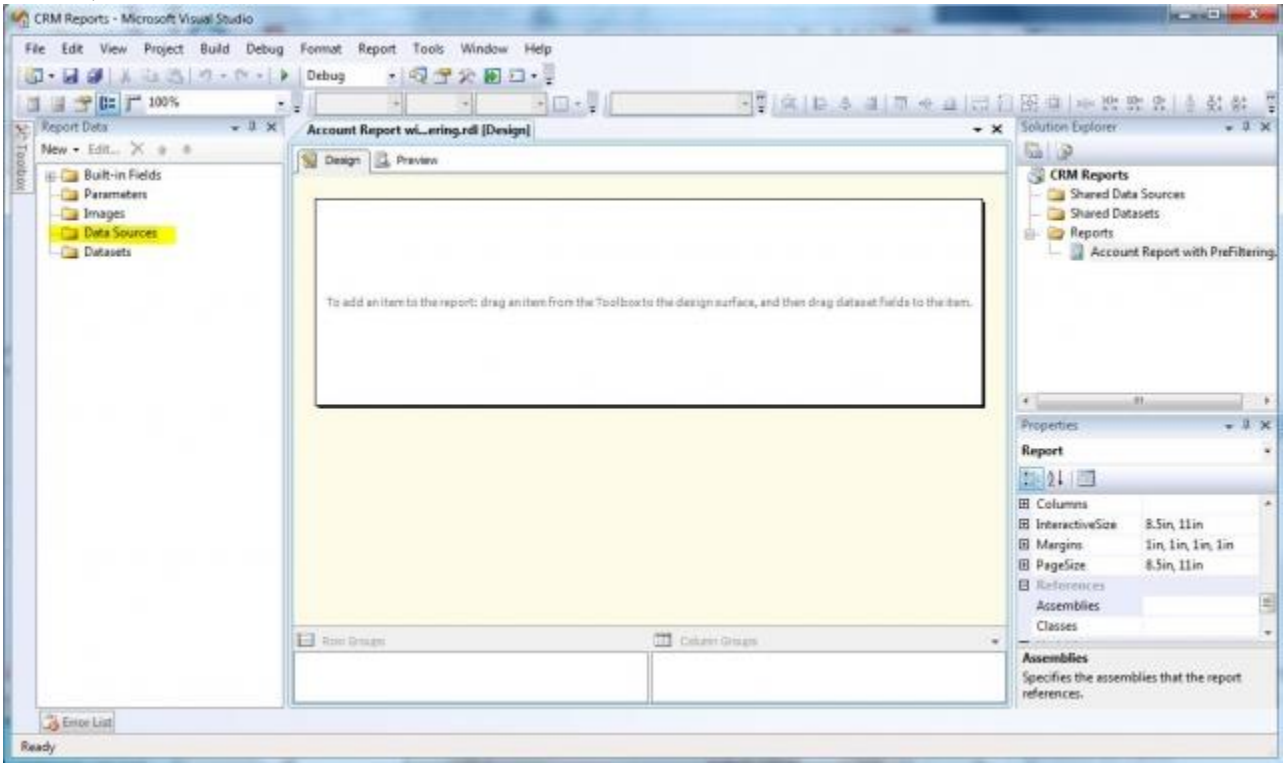
Next, we will create the report. To do this, right click on the Reports folder in the Solution Explorer and select **Add >> New Item** (do not choose Add New Report as this will create the report with the Report Wizard).



From the **Add New Item** window, select the Report Template, give your Report a name (I am using Account Report with PreFiltering) and click **ADD**.

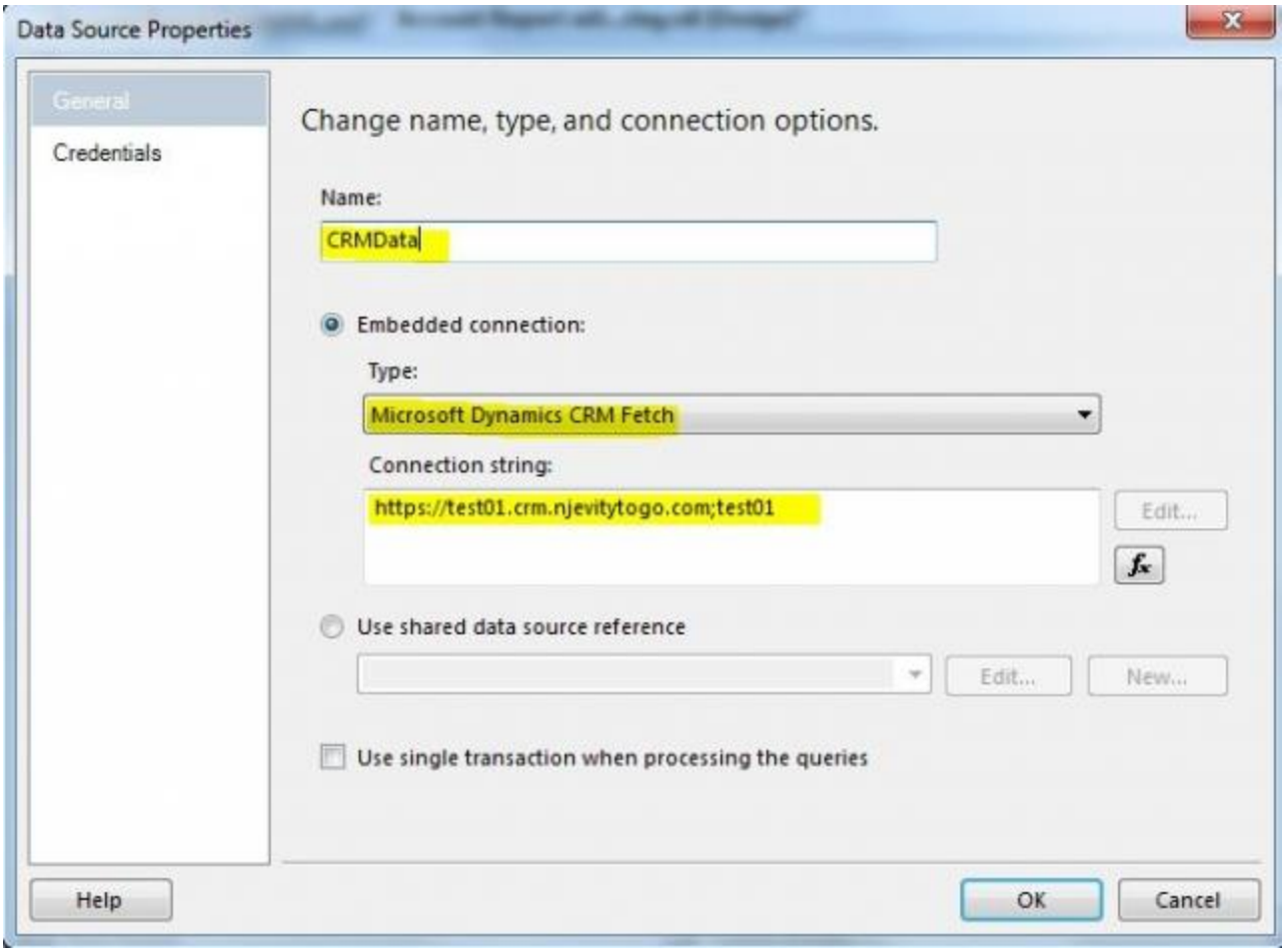


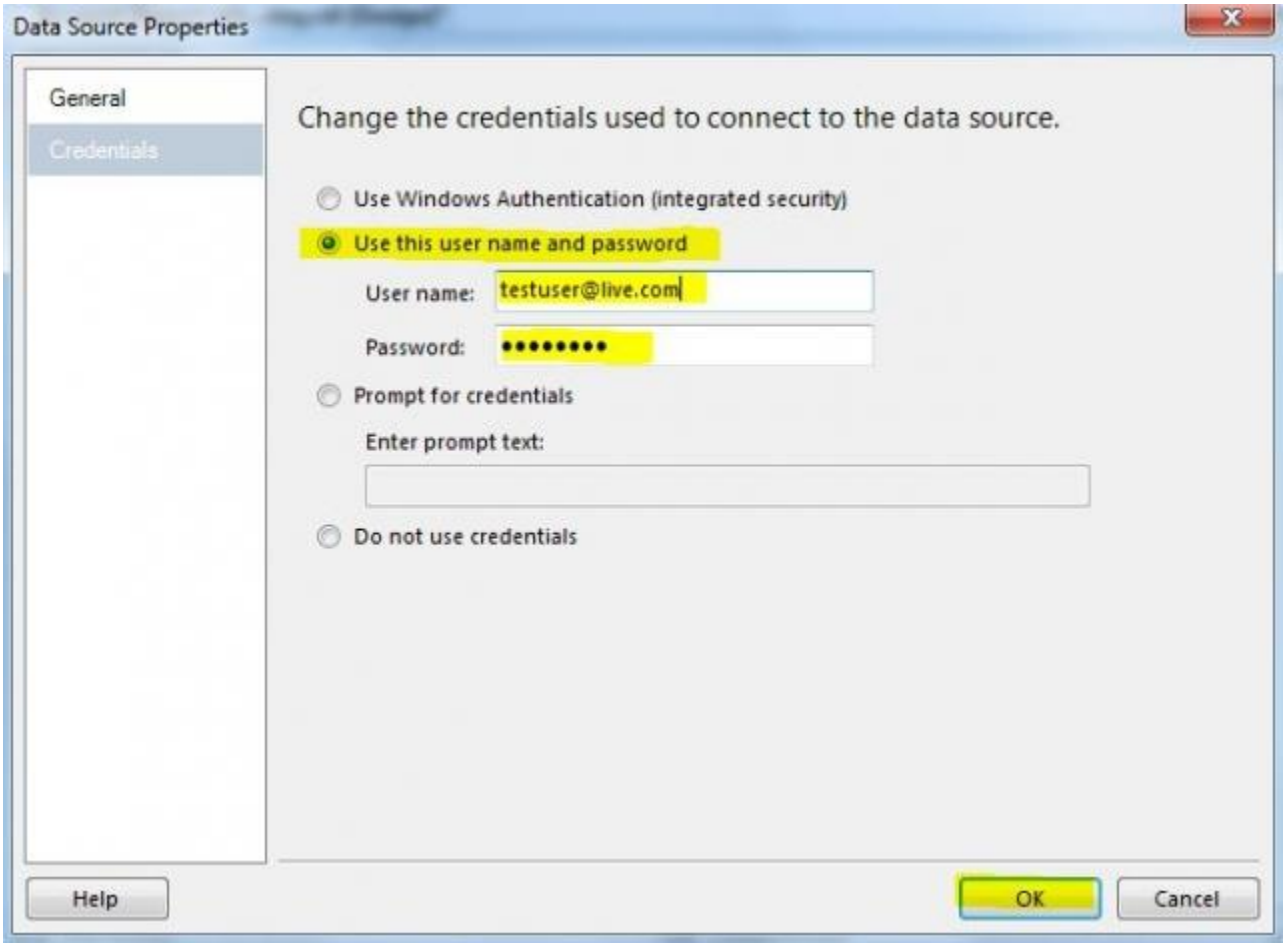
Next, we need to create the data source for the report. With Dynamics CRM 2011 Reports, you should use a Data Source embedded in the report, not a shared data source. To do this, in the Report Data pane on the left side of Visual Studio, right click on the Data Sources Folder and choose, **Add Data Source.**



In the Data Source Properties window:

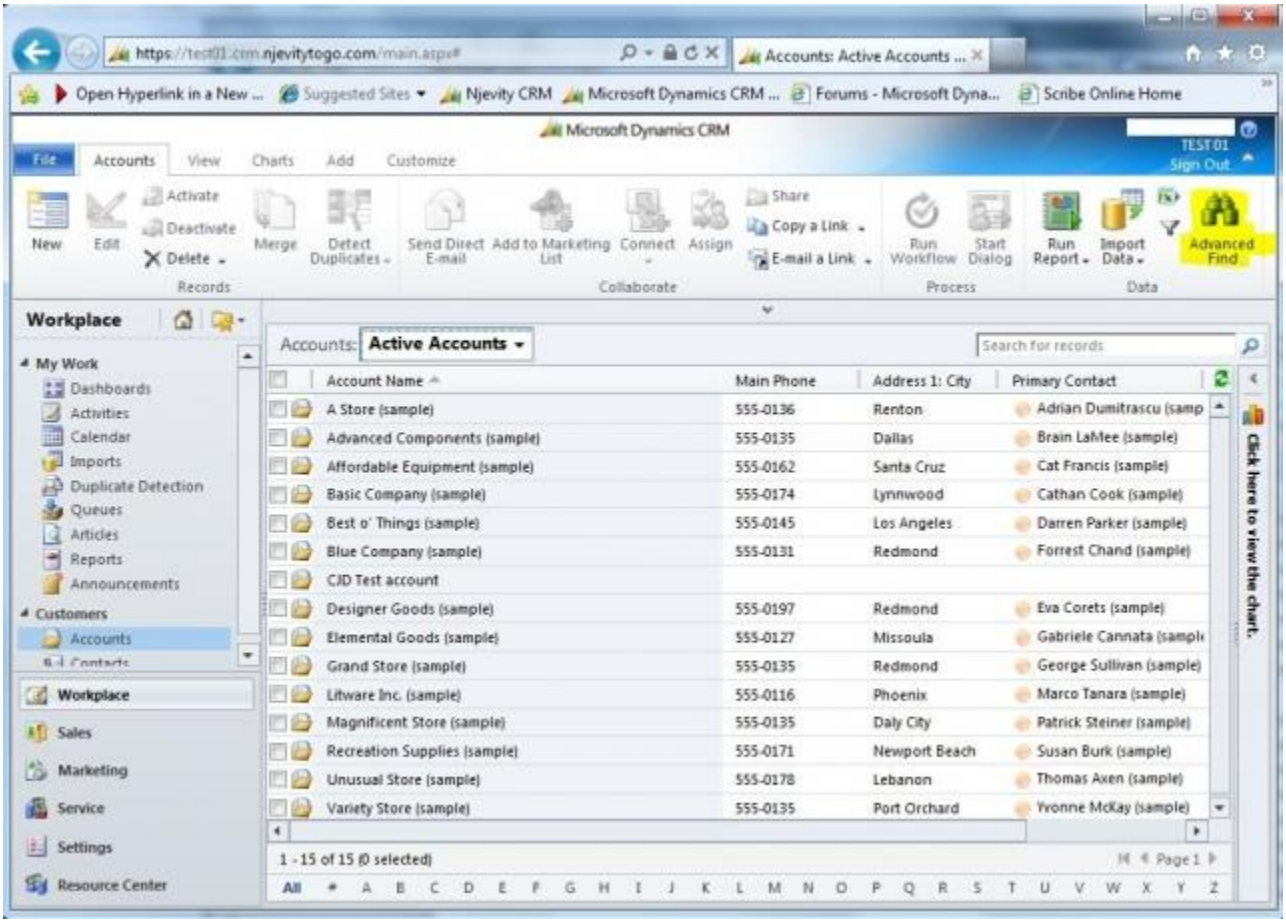
1. Give the Data Source a Name (I am using CRMData)
2. Select Embedded Connection of Type: **Microsoft Dynamics CRM Fetch**
3. In the connection string box, enter the url of your CRM Organization folowed by a semicolan and your organization unique name. You should be able to copy and paste the first part of this from your web browser. I am using https://test01.crm.njevitytogo.com;test01
4. Click the **Credentials** tab
5. Select '**Use this user name and password**' and enter the user credentials that you use when connecting to your CRM Organization.
6. Then click **OK**


Data Source Properties
General
Credentials
Change name, type, and connection options.
Name:
CRMData
Embedded connection:
Type:
Microsoft Dynamics CRM Fetch
Connection string:
https://test01.crm.njevitytogo.com;test01
Edit...
Use shared data source reference
Edit...
New...
Use single transaction when processing the queries
Help
OK
Cancel

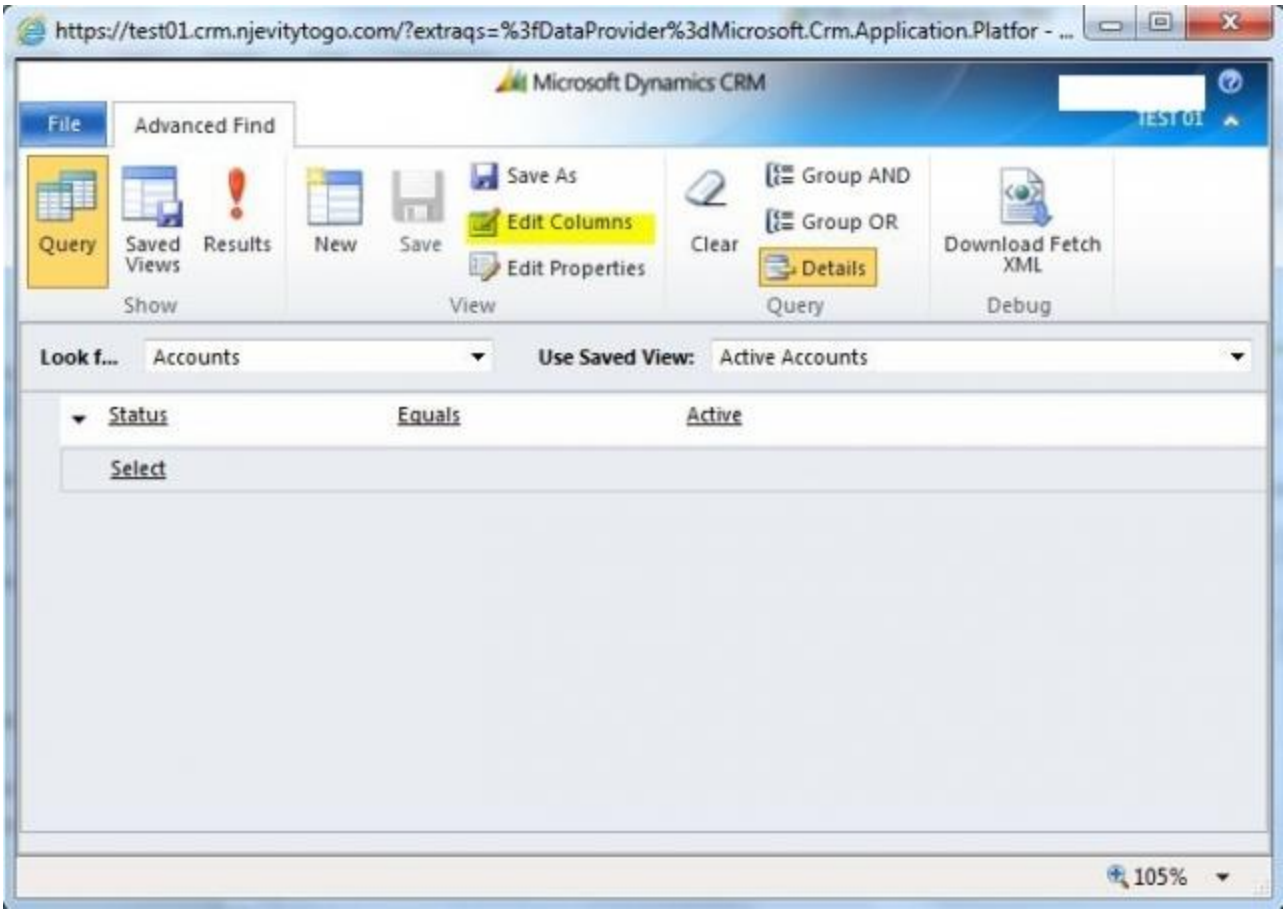


Next, we need to create the dataset for the report. Because we chose to use CRM Fetch in the previous step, we will be using FetchXML as the query language. Fortunately, for those of us who do not know how to write FetchXML queries, there is a very simple way to do this: we will let Advanced Find create the FetchXML for us!

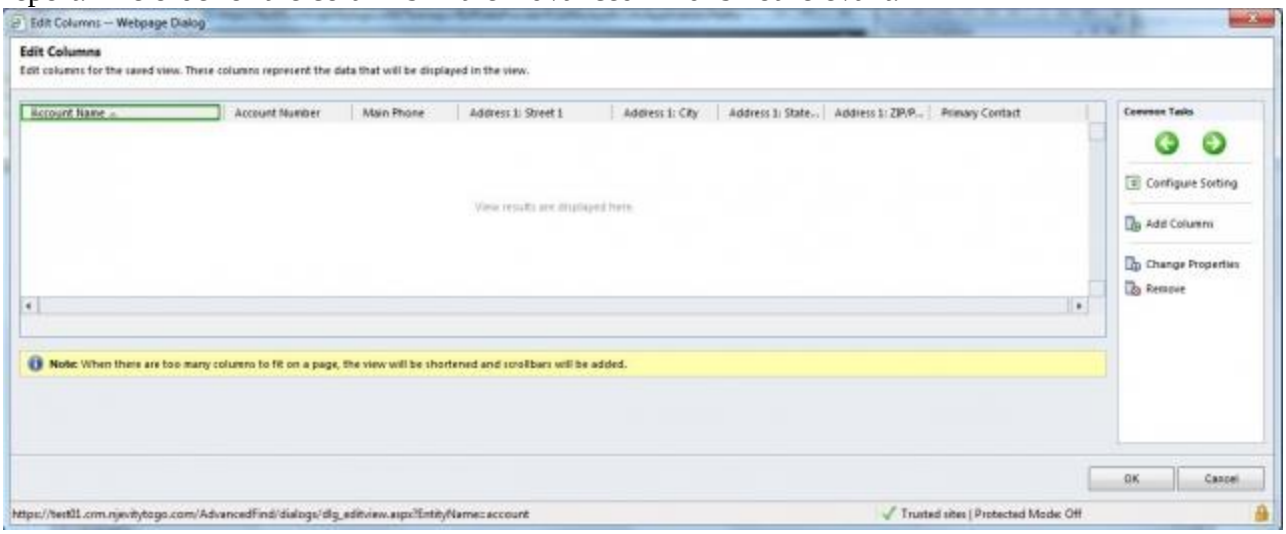
To do this, we will create an Advanced Find that contains the columns we want on our report as well as the filters that we want always applied to the report. For this example, we will create an advanced find that returns ACTIVE Accounts and displays a few relevant columns. So, in Dynamics CRM 2011, navigate to an account list and click the **Advanced Find button**.



Because I had the Active Accounts view displayed on the Account List when I opened Advanced Find, the Advanced Find query automatically opens with the **Status = Active** filter applied. Because we ONLY want active accounts and we do not want the user to be able to edit this filter, we will keep this filter in place. We could add additional filters here, just keep in mind that the *users cannot change these filters when they run the report*.

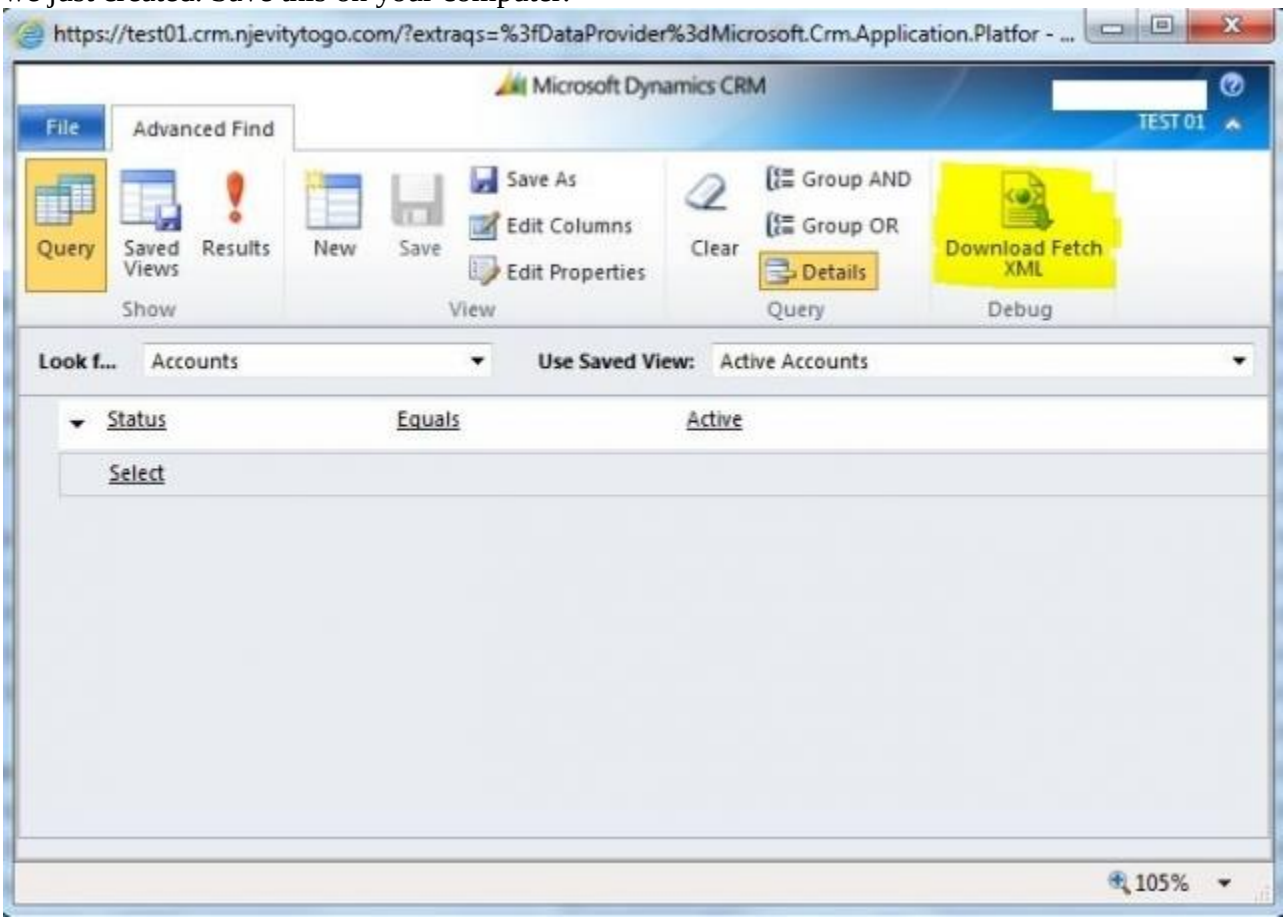


Next we will select the columns that we want to work with on our report. To do this, click the **Edit Columns** button and add and remove columns until you have the ones that you want on the report. The order of the columns in the Advanced Find is not relevant.

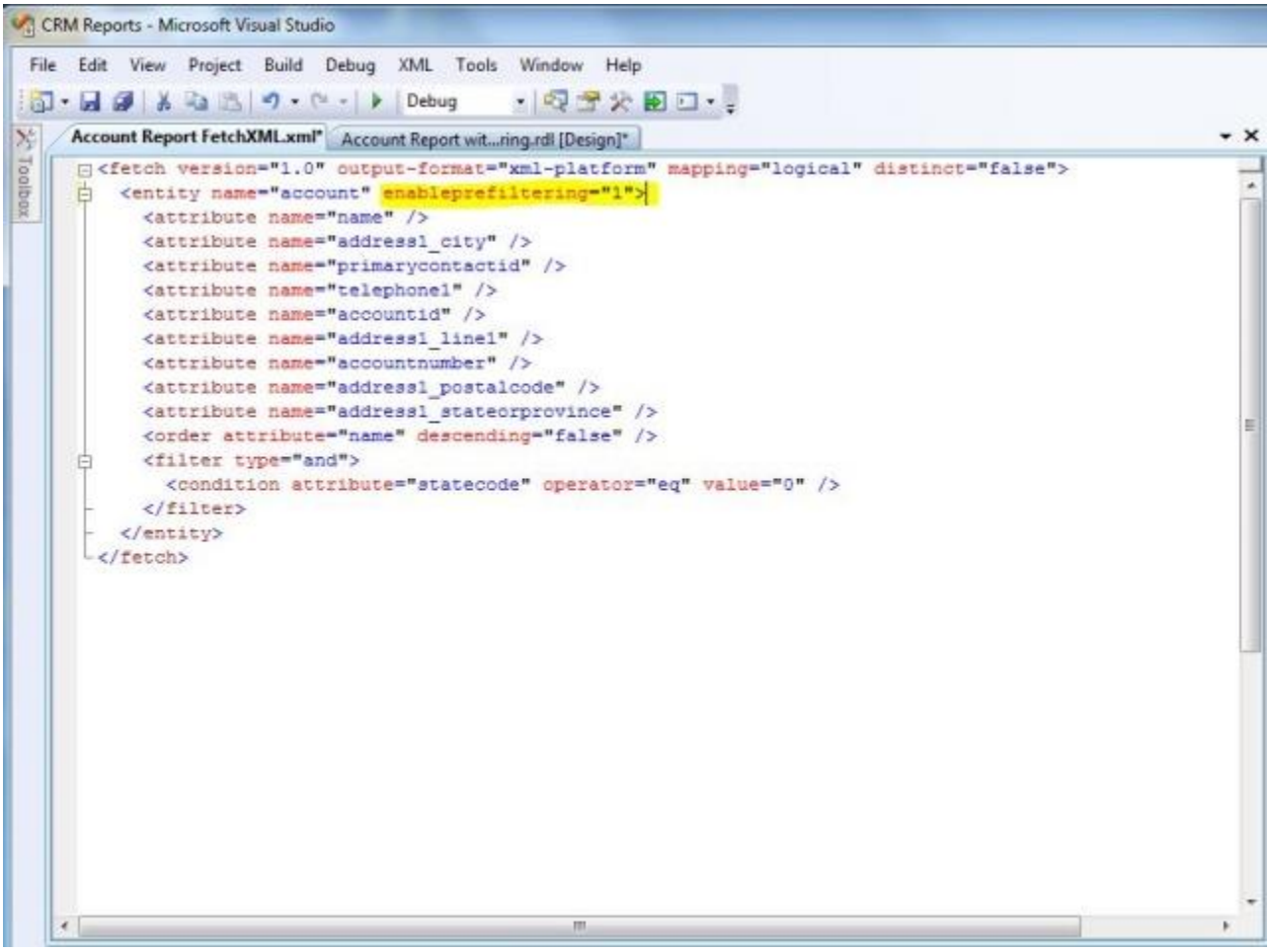


Click the **OK** button to close the Edit Columns window and return to the Advanced Find window. At this point, you might want to run the Advanced Find to make sure that the data set looks right. Once you are comfortable with the dataset, click the **Download Fetch XML** button

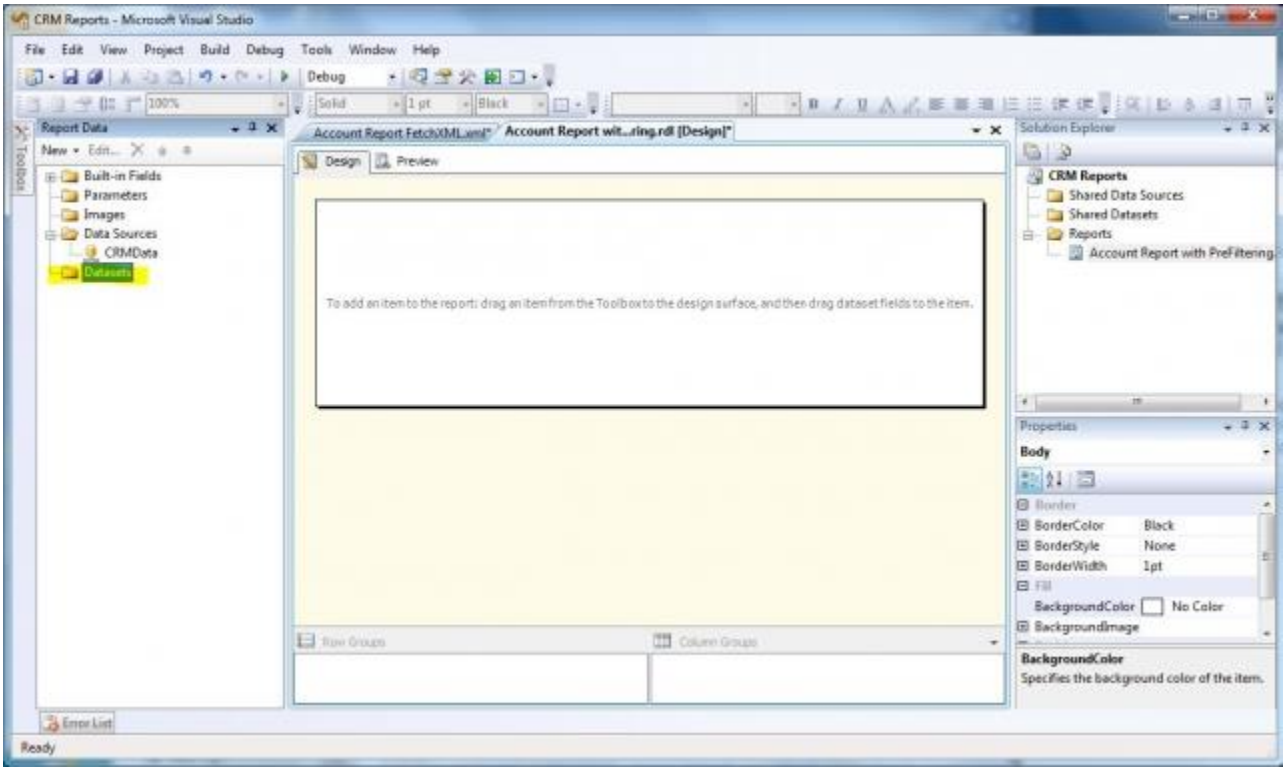
in the Advanced Find Ribbon. This will download an xml file that contains the FetchXML query we just created. Save this on your computer.



Then locate the file, right click and open the file in Visual Studio. It should look like the file below. Now, add the **enableprefiltering** parameter and set its value to "1" as indicated below. This will enable Pre-Filtering and Default Filters on the report. If you do not add this parameter, you can still run the report in Dynamics CRM 2011, but you will not be able to apply filters to it in Dynamics CRM 2011.

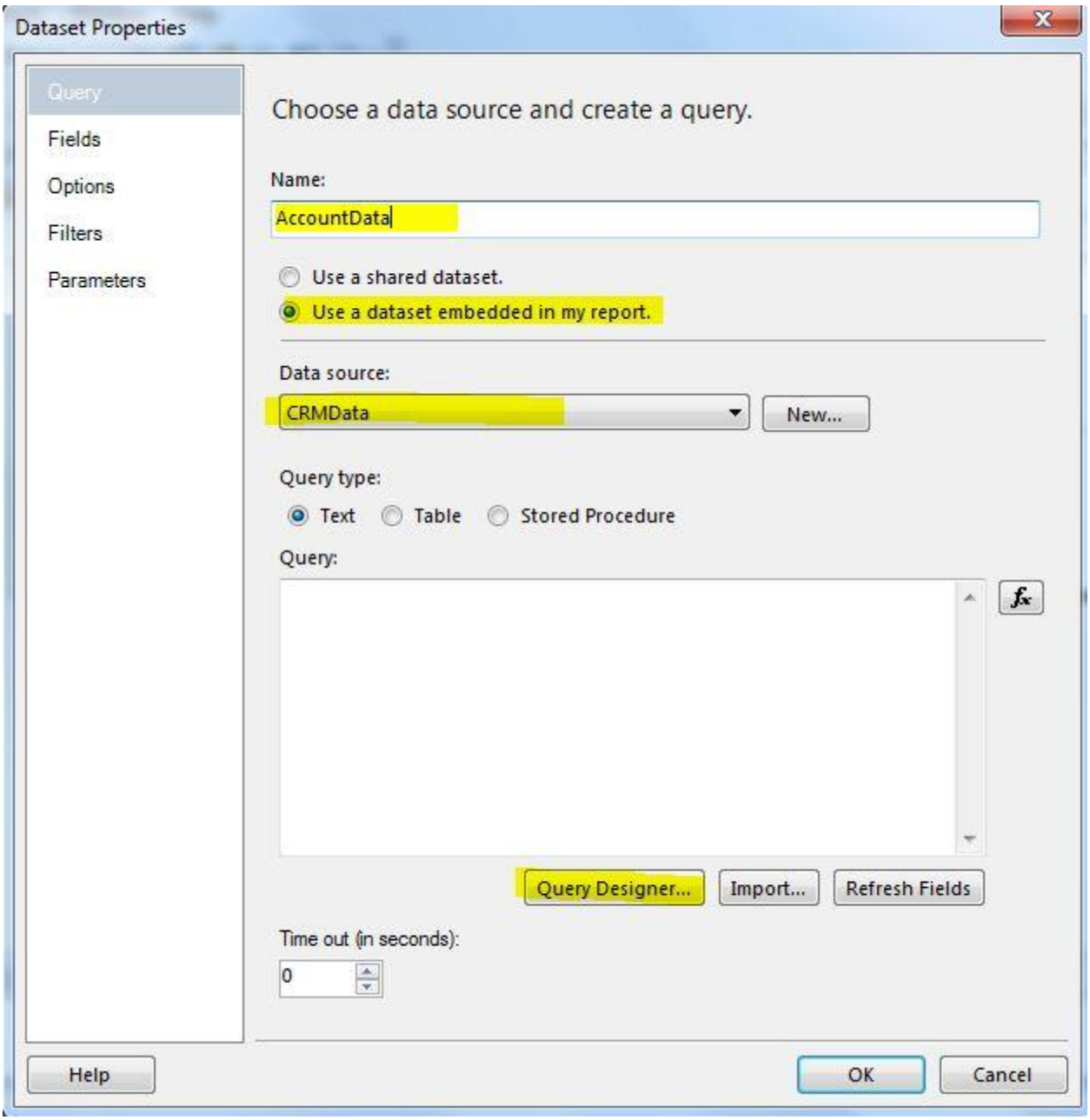


Next, copy the xml to your clipboard and return to the report in Visual Studio. Right click the **Datasets** folder in the Report Data pane on the left side of the Visual Studio window and select **Add Dataset**.

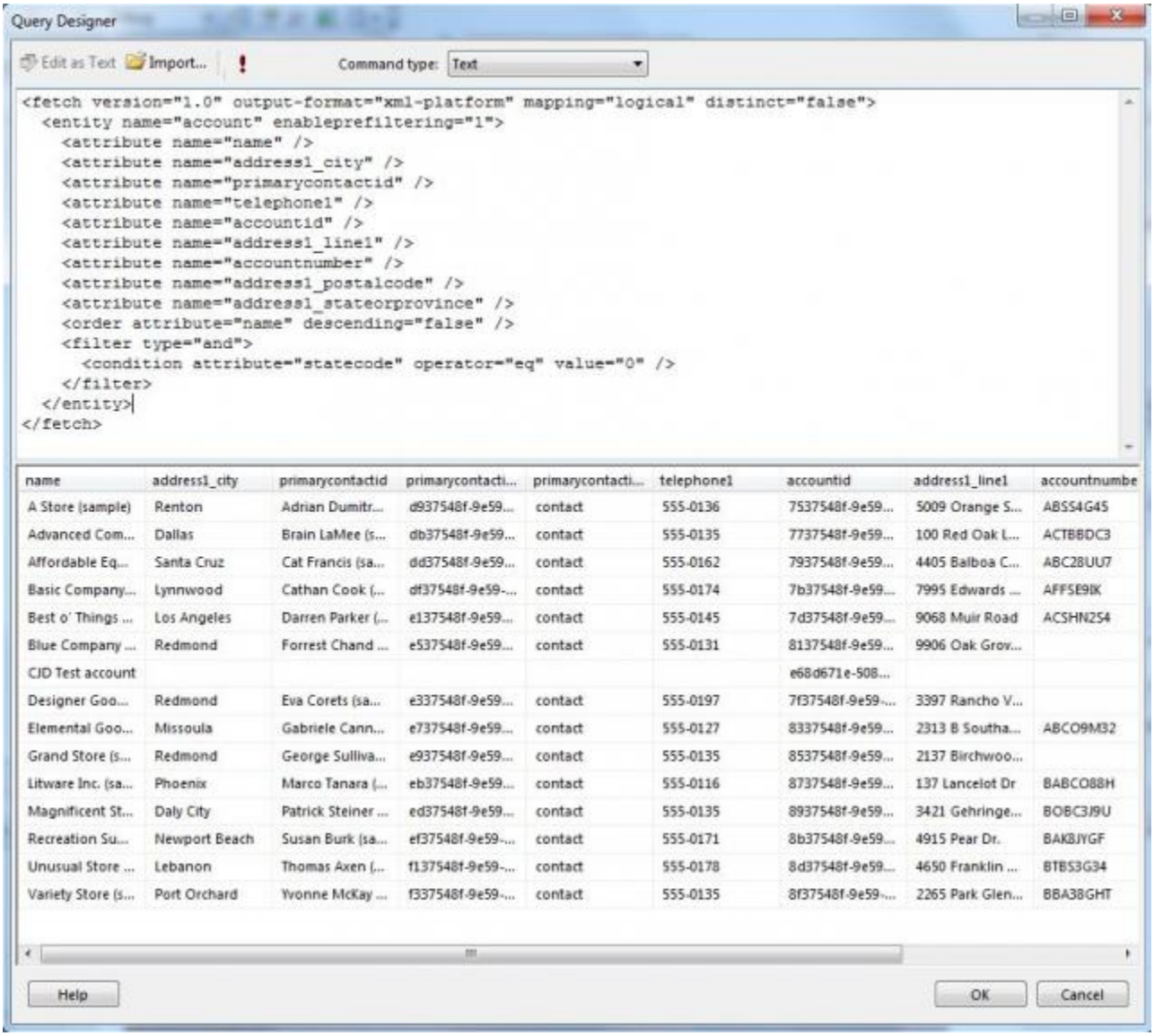


On the Dataset Properties window:

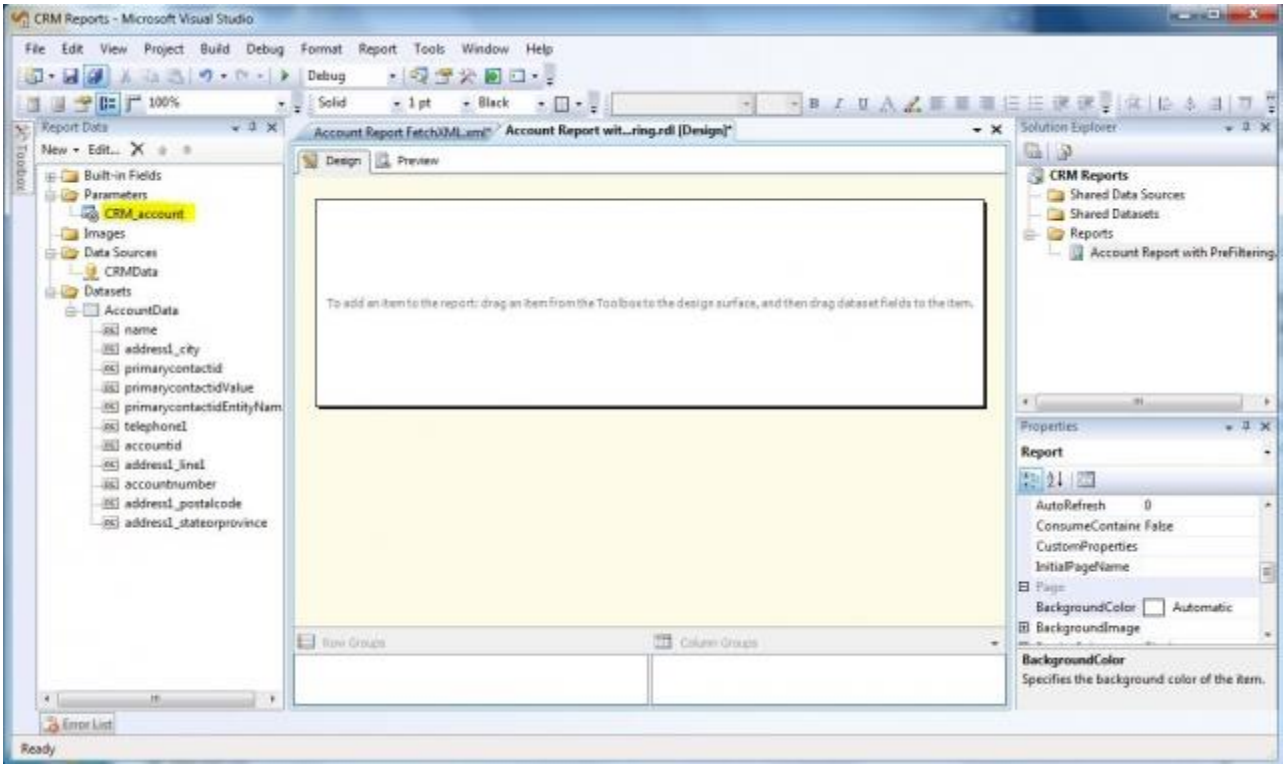
1. Give your dataset a name. I am calling mine AccountData.
2. Select the radio button labeled '**Use a dataset embedded in my report**'.
3. In the Data Source drop down, select the data source you created in a previous step.
4. Click the **Query Designer** button



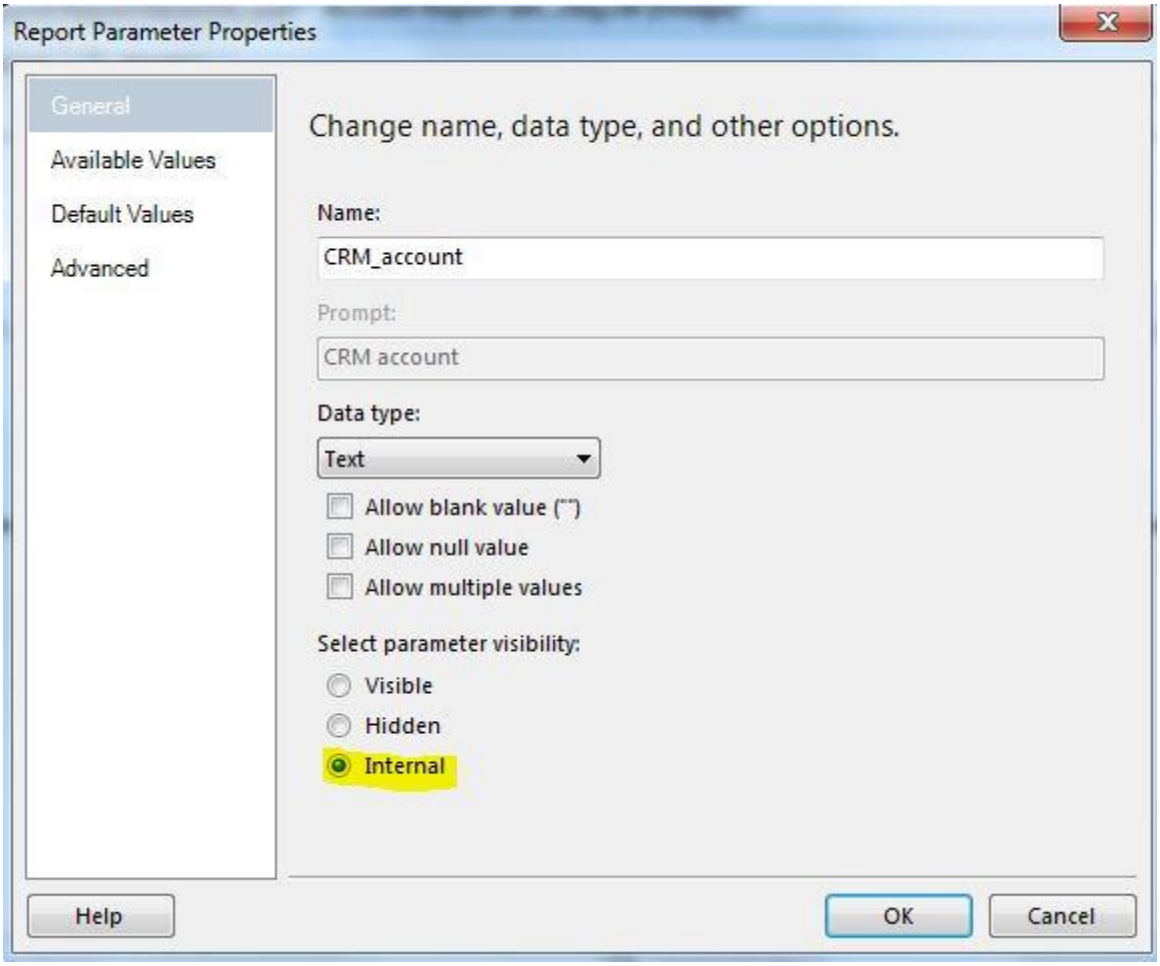
In the text box at the top of the query designer, paste the FetchXML you copied from the XML File. Then click the **Red Exclamation Point** button to execute the query. When the Define Query Parameters window opens, leave the parameter value for the **CRM_account** parameter blank and click OK. This will return the same set of records that were in the advanced find query.



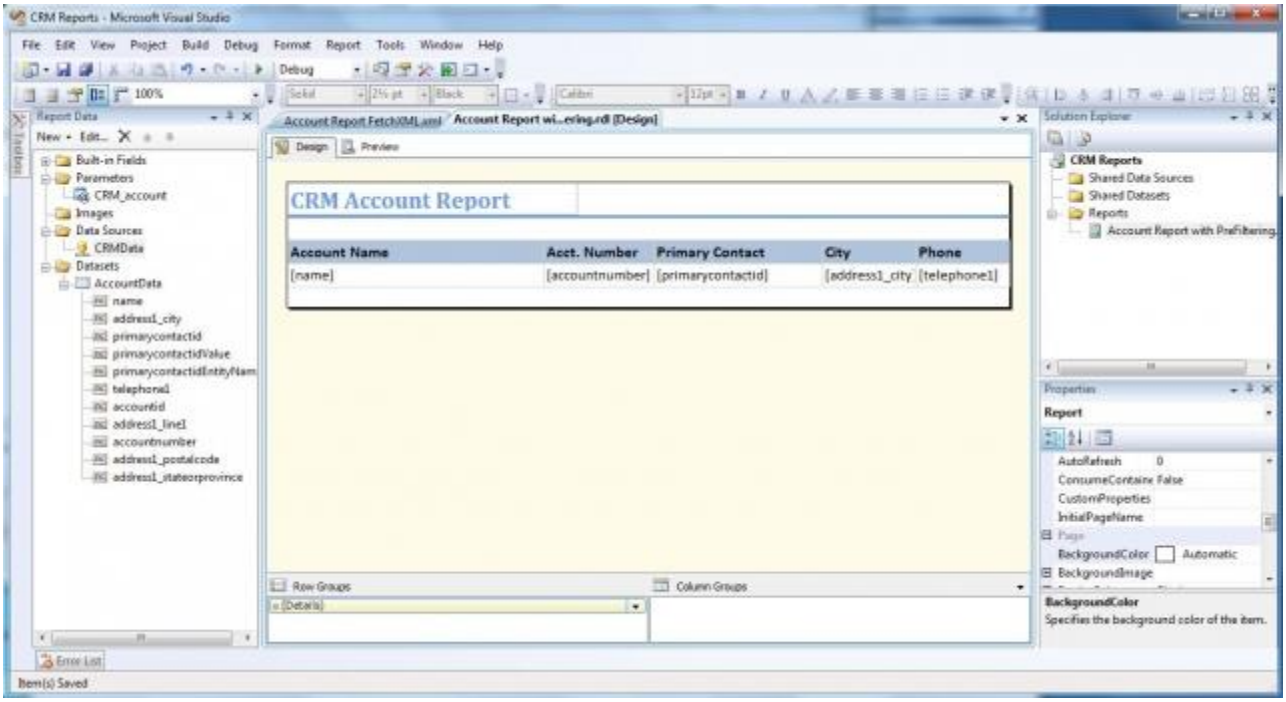
If you are happy with the results, click **OK** to close the **Query Designer,** then click **OK** to close the **Dataset Properties** window. If you expand the Dataset, you should see the list of fields just like below. Next, we need to edit the Parameter that was automatically created for the Pre-Filtering. In this case, the parameter is called CRM_account. To edit this, right click the parameter and select **Parameter Properties**.



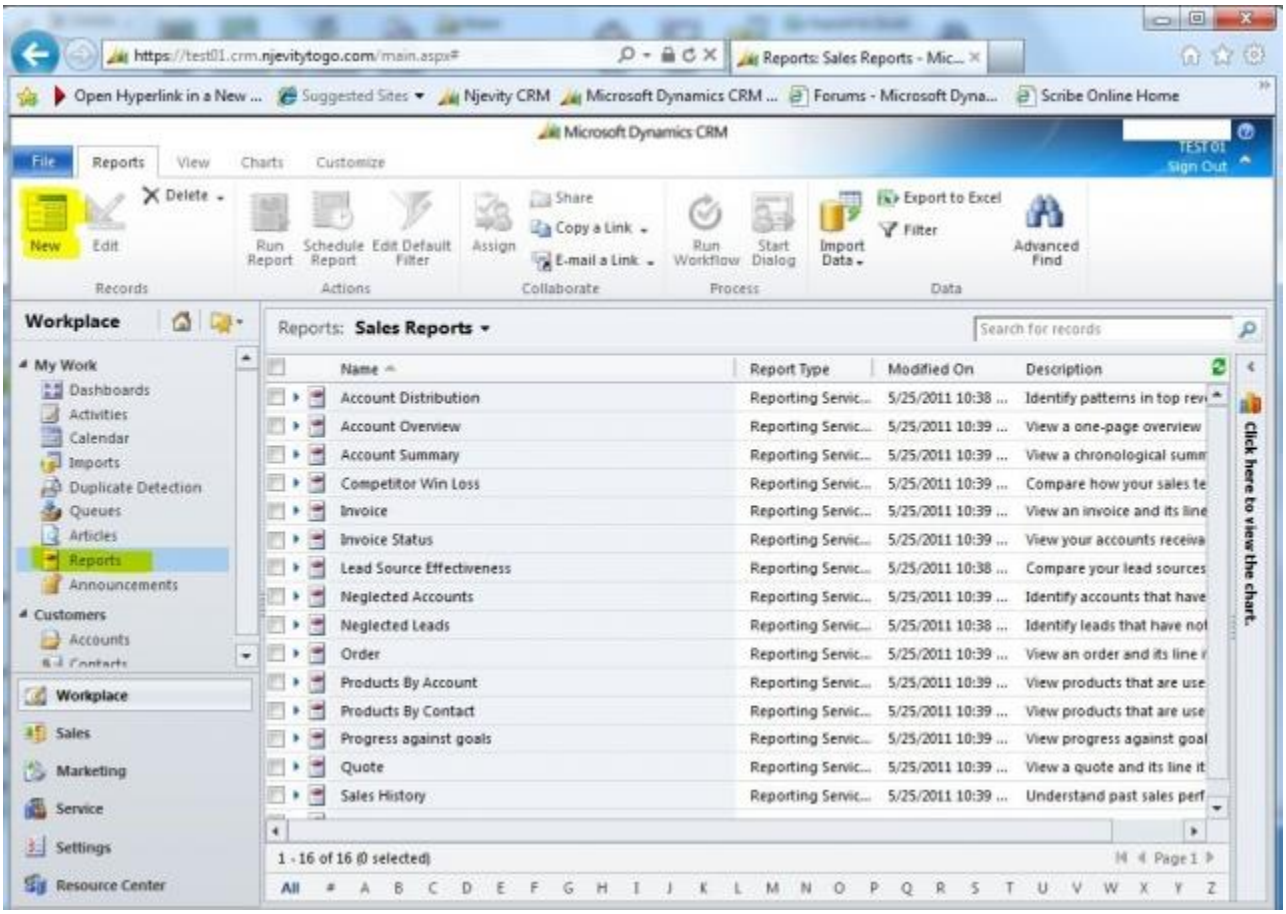
On the Report Parameter Properties window, select the radio button labeled '**Internal**'. This will prevent the user from being prompted for a value for this parameter when they run the report. Since we are getting this paramater value directly from CRM, there is no need for the user to interact with this parameter. Click **OK** to close this window.



At this point, you can design this report just like you would any other SRS Report. In this example, I have added a title and table to the report and put a few of the fields into the table. When you are satisfied with the report, **save** your changes.

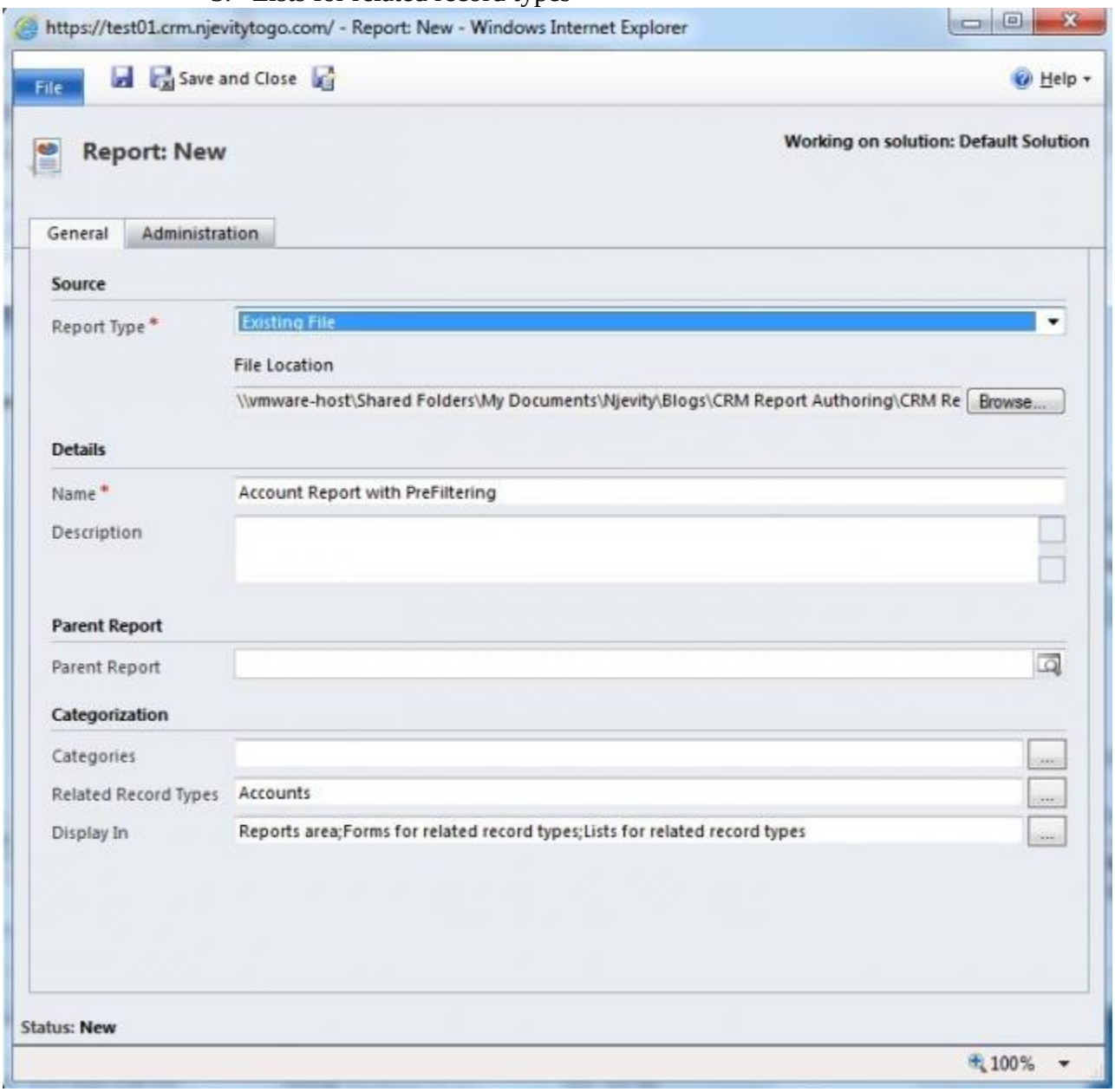


Now, we need to upload the report to Dynamics CRM 2011. To do this, we will log in to Dynamics CRM 2011 and navigate to the Report List (**Workplace >> Reports**) and click the **NEW** button.



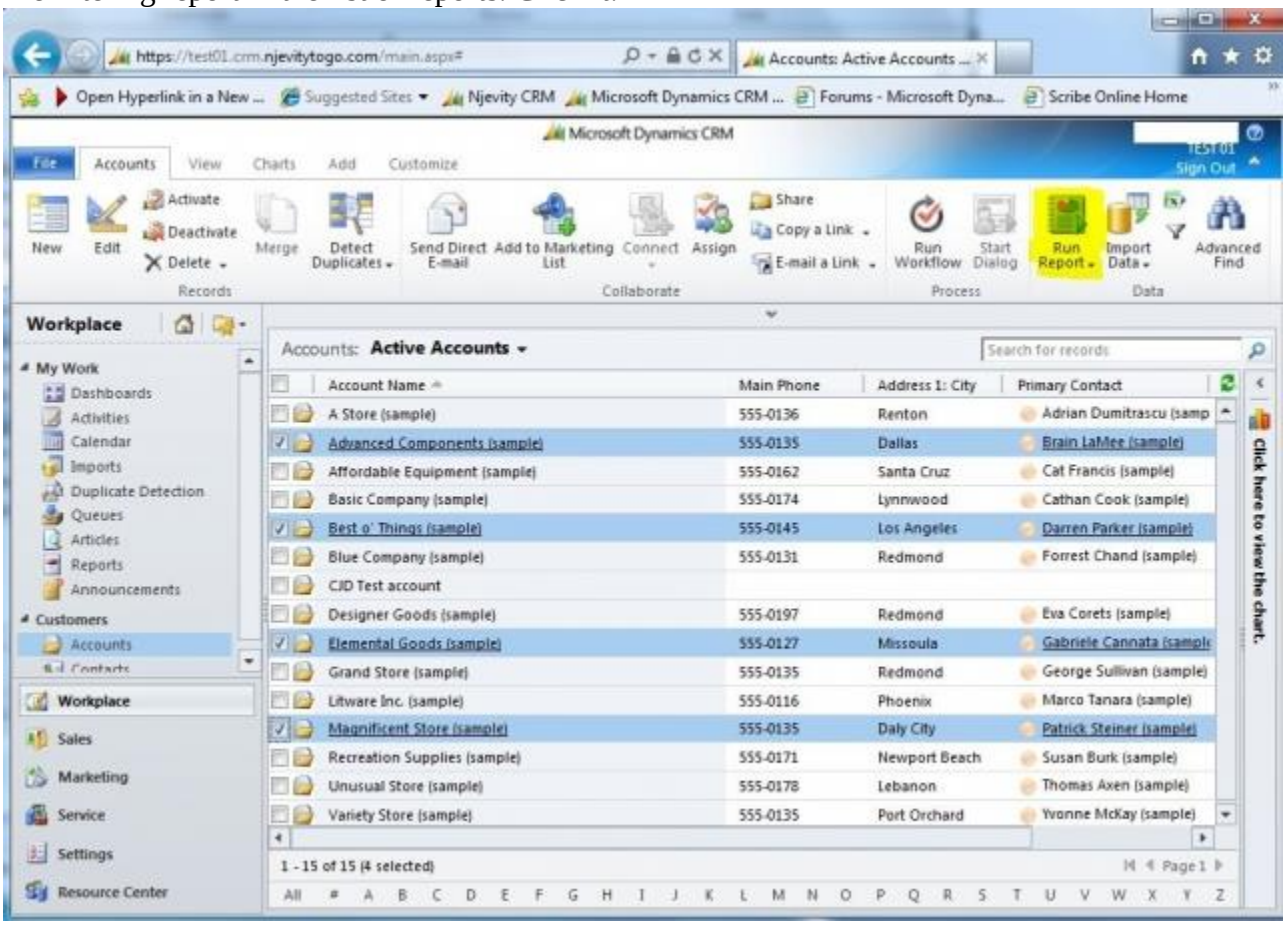
From the new report window:

1. Report Type: Existing File
2. File Location: Navigate to the RDL file that you created in Visual Studio. Mine is called 'Account Report with PreFiltering.RDL'
3. Name: This defaults from the RDL file name, but you can change it if you wish
4. Related Record Types: This is the CRM Entity the report is linked to. In this case, Accounts
5. Display In: Select:
    1. Reports Area
    2. Forms for related record types
    3. Lists for related record types

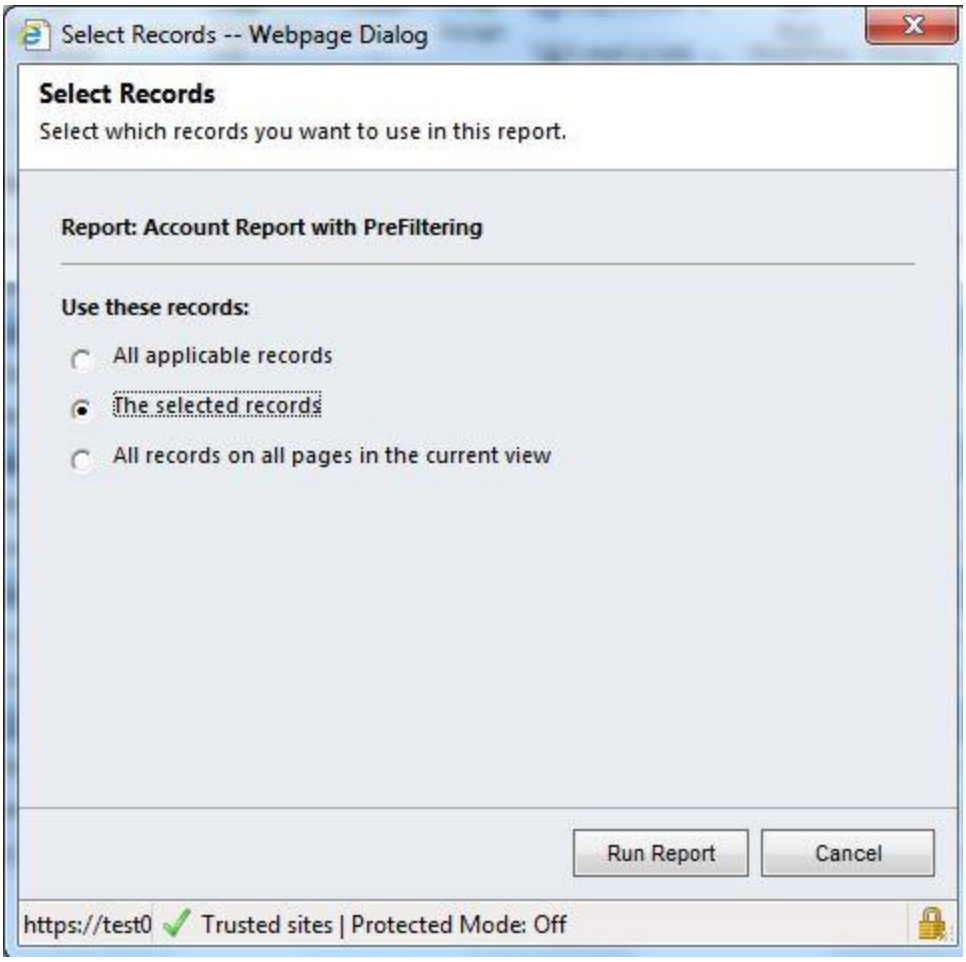


Once this is done, click **SAVE AND CLOSE** to upload and save the report to Dynamics CRM 2011.

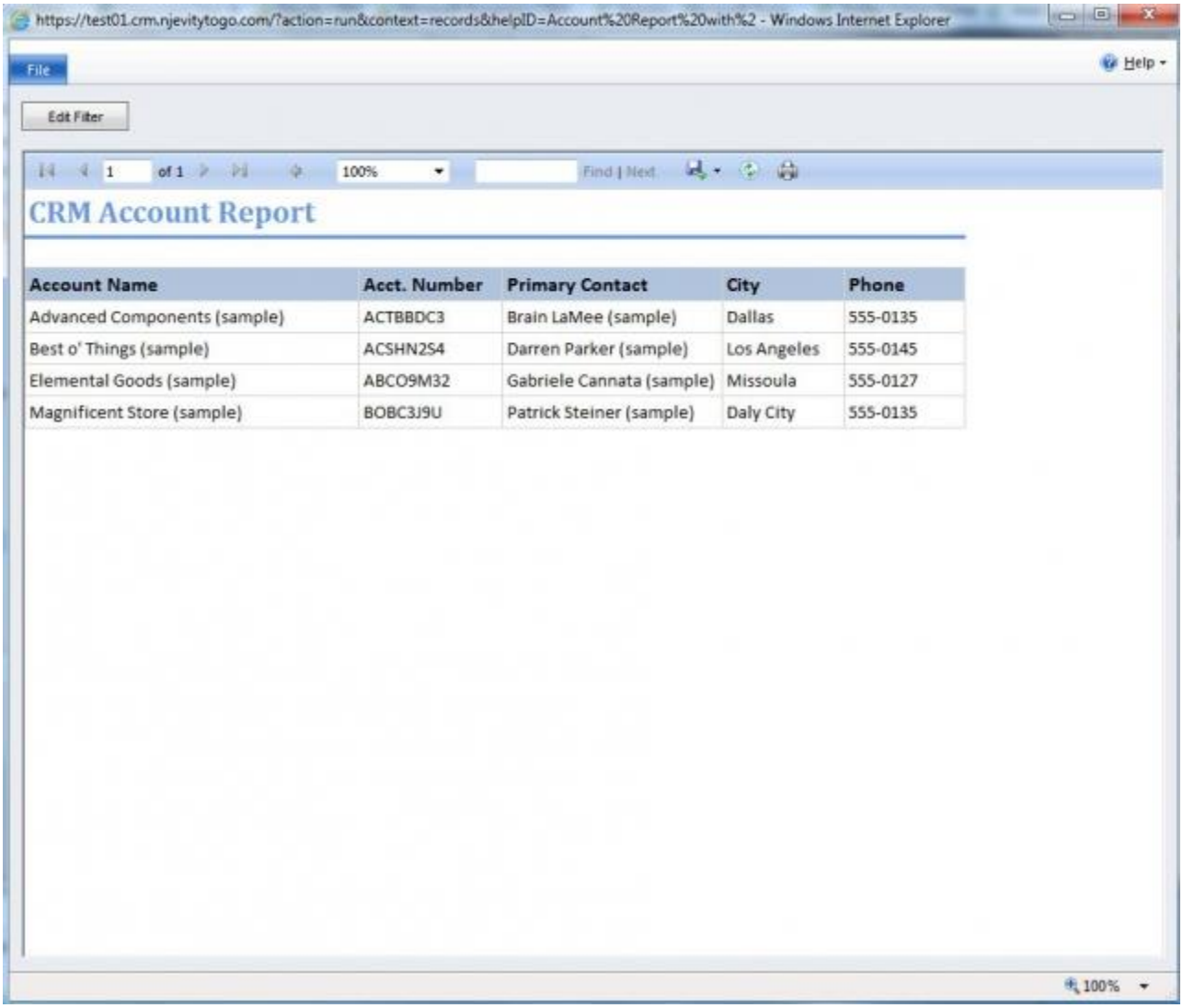
Now, lets go run the report. Navigate to the Account List. Select a few accounts and click the **Run Report** button in the Account Ribbon. You should see the Account Report with PreFiltering report in the list of reports. Click it.



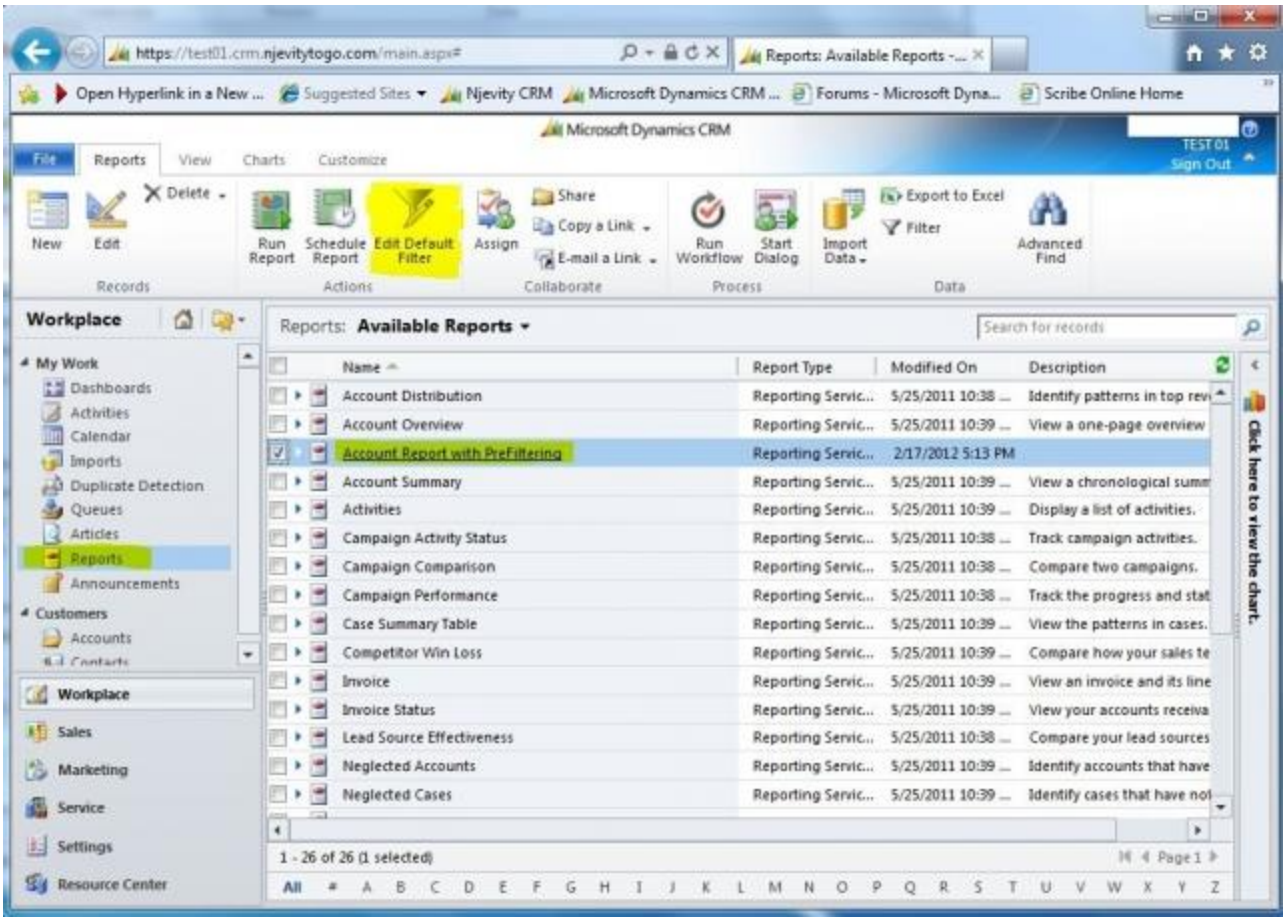
Choose to run the report on **the selected records**.



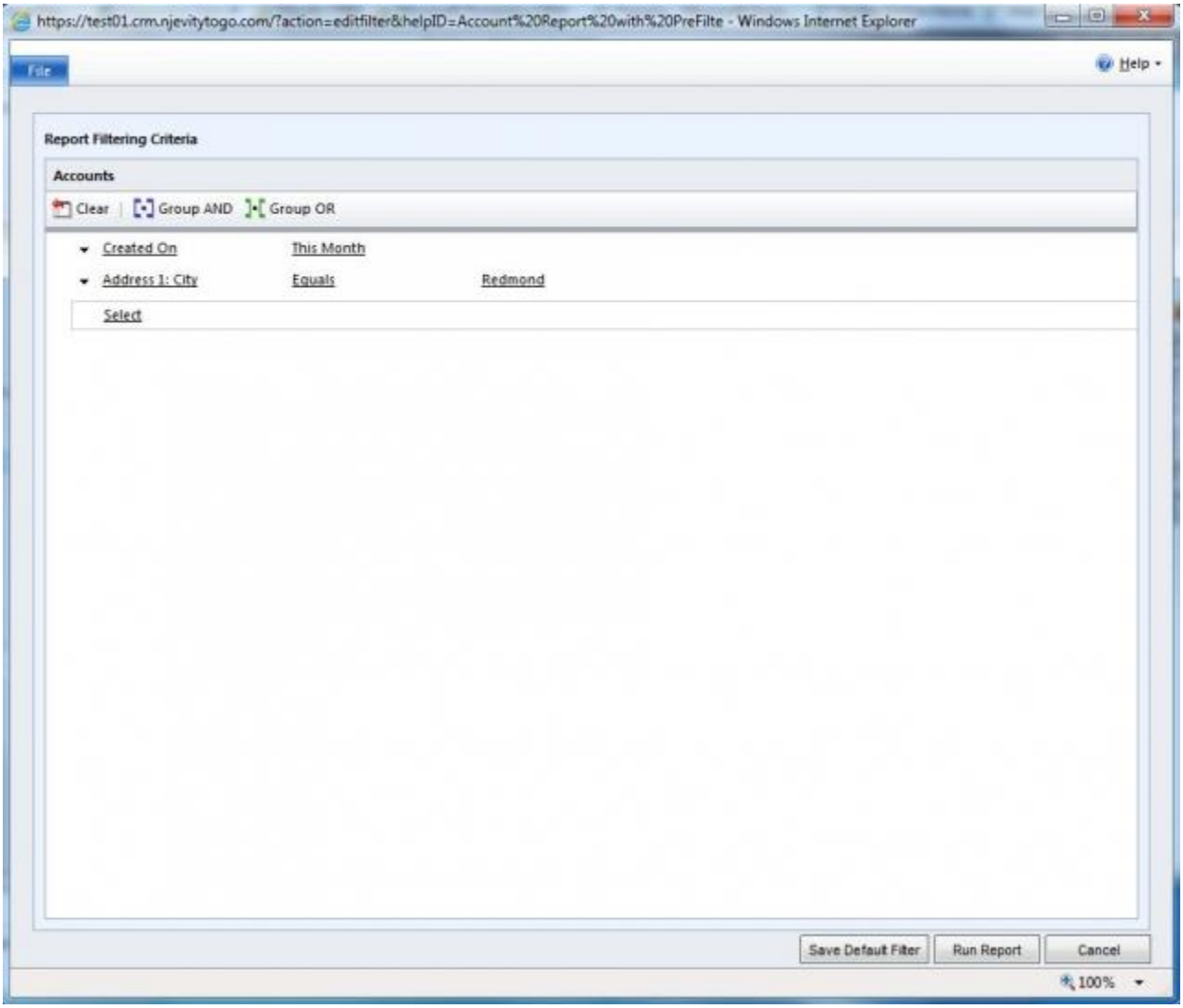
As you can see, the only Accounts that are included on the report are the ones that you selected.
This is a successful implementation of Pre-Filtering on the report.



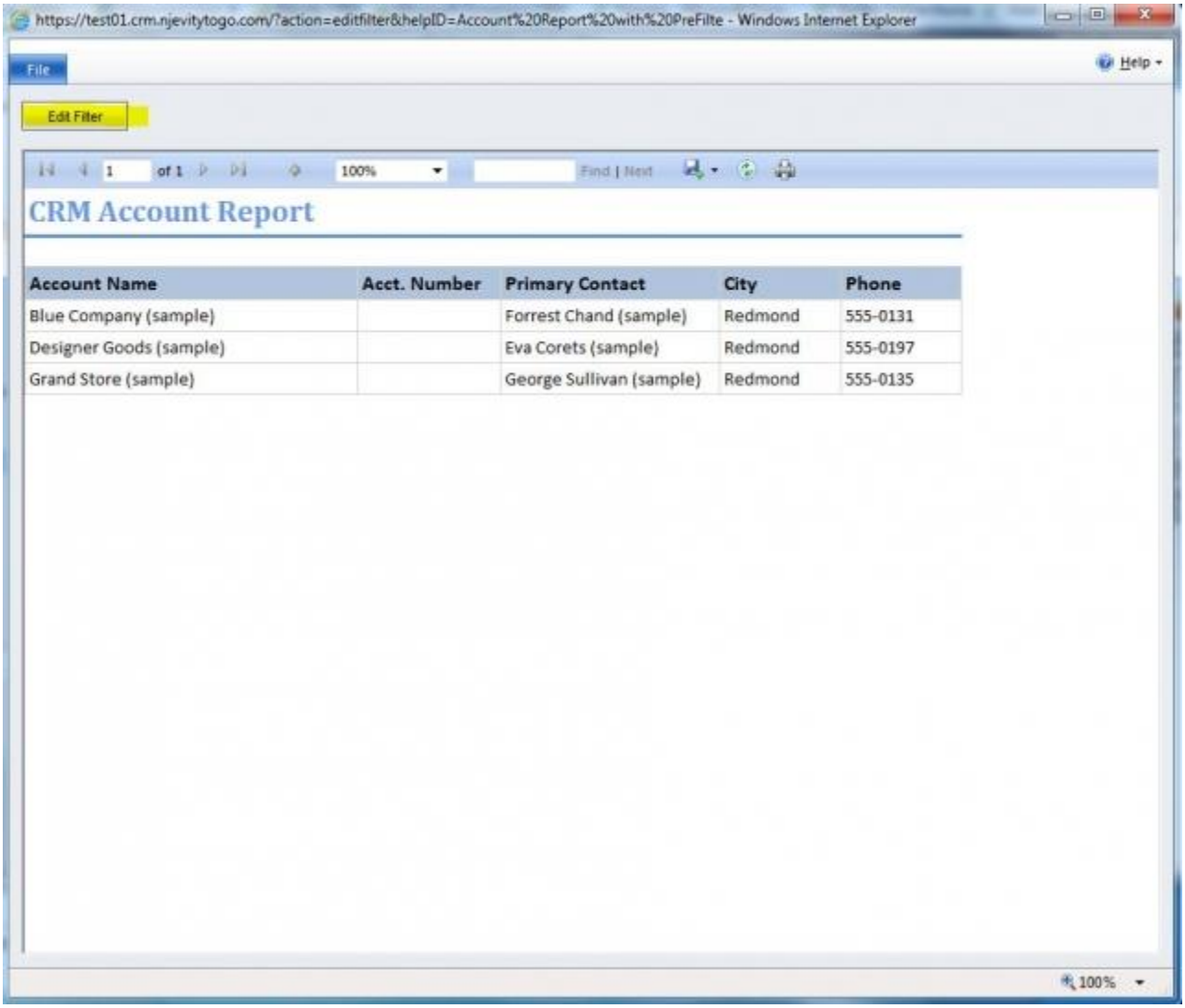
Now, lets look at setting a default filter on the report. To do this, return to the Reports List (**Workplace >> Reports**) and find the **Account Report with PreFiltering** report on the list. Select it and click the **EDIT DEFAULT FILTER** button in the Reports List Ribbon.



On the Report Filter Criteria window, enter your filters just like you would in Advanced Find. Notice that you have access to every field and every relationship in CRM--not just the ones included in the report. In this case, I am selecting to only display Accounts that were created This Month and are in the City of Redmond. Click the **SAVE DEFAULT FILTER** button to save this as the default filter on the report. When you do this, this will be the filter applied by default when you run the report from the Reports List or from the Account List when you select to run the report on applicable records. If you run the report on the selected records as we did above, that filter will replace the default filter.



Users may change the filter each time they run the report by clicking the **Edit FILTER** button in the top left corner of the report.



This is how you can create a report and allow the users to specify any filters (parameters) when they run the report. And as you can see, making these changes does not require any report development work. It can all be done by the end user. Simple. Easy. Hard not to love it!