

Enesys RS Data Extension Manual 2.0

Reporting Services Data Processing Extension for SharePoint

ENESYS

Enesys RS Data Extension Manual

Reporting Services Data Processing Extension for SharePoint

Sommaire

- ABOUT ENESYS RS DATA EXTENSION 4
 - LICENSE AGREEMENT 4
- INSTALLATION AND UPGRADE 6
 - SYSTEM REQUIREMENT 6
 - INSTALLATION 6
 - OVERVIEW 6
 - REPORT SERVER INSTALLATION 7
 - BUSINESS INTELLIGENCE DEVELOPMENT STUDIO INSTALLATION 10
 - SAMPLE CONFIGURATION FILES 11
 - INSTALLING SCHEMA FILES 12
 - UPGRADING FROM VERSION 1.X 12
 - OVERVIEW 12
 - UPGRADING 13
- QUICK START 15
 - CREATING A NEW REPORT PROJECT 15
 - ADDING A NEW REPORT 16
 - CREATING A DATASET 16
 - CREATING A QUERY 18
 - LAYOUT 19
 - CONCLUSION 19
- USING ENESYS RS DATA EXTENSION 21
 - RETRIEVING DATA FROM A SHAREPOINT LIST 21
 - SPECIFYING SHAREPOINT LIST 22
 - SPECIFYING LIST COLUMNS 22
 - FILTERING LIST DATA 22
 - USING COLUMN DISPLAY NAMES OR INTERNAL NAMES 23
 - USING REPORT PARAMETERS 24
 - PENDING ITEMS 26
 - EXPANDING RECURRING EVENTS 26
 - STRIPPING HTML TAGS 29
 - USING RUNNING VALUES 31
 - RETRIEVING ITEMS AND FOLDERS IN A SPECIFIC FOLDER 33
 - RETRIEVING THE FIRST N ITEMS OF A LIST 34
 - CUSTOM SEPARATOR MULTIPLE VALUES COLUMNS 35
 - EXPANDING MULTIPLE VALUES 36
 - APPLYING OPERATIONS TO LISTS 37
 - JOINING RESULT SETS 38
 - MERGING RESULT SETS 39
 - GETTING DISTINCT VALUES 40
 - GETTING A SUBSET OF A RESULT SET 41
 - MERGING MULTIPLE LISTS 42
 - MERGING LISTS USING A REFERENCE LIST 42
 - ROLLING UP LISTS IN A SITE COLLECTION 44
 - RETRIEVE LISTS INFORMATION, GROUPS AND PERMISSIONS 46

LIST COLLECTION	46
LIST PERMISSIONS.....	47
WEB PERMISSIONS.....	47
SHAREPOINT GROUPS.....	47
SPECIFYING A SET OF SITES	48
RETRIEVING ENESYS RS DATA EXTENSION VERSION INFORMATION	48
SAMPLE REPORTS.....	49
FILES DETAILS.....	49
SAMPLE REPORTS.....	49
DATA SOURCE CREDENTIALS.....	51
REPORT DESIGNER	52
REPORT SERVER.....	53
WHICH CREDENTIALS SHOULD YOU USE?	55
USING INTELLISENSE FOR WRITING QUERIES.....	55
REFERENCE.....	59
LIST ELEMENT	59
FIELDS ELEMENT	61
QUERY ELEMENT	61
CUSTOMFIELDS ELEMENT	61
MULTILIST ELEMENT	62
MERGING LISTS APPROACH	62
ROLLING UP LISTS APPROACH.....	63
MERGING LIST COLUMNS	63
SQLOP ELEMENT.....	64
OP= "JOIN"	64
OP= "OUTERJOIN"	64
OP= "UNION"	64
OP= "DISTINCT"	65
OP="SELECT"	65
LISTCOLLECTION ELEMENT	65
LISTPERMISSIONS ELEMENT.....	66
WEBPERMISSIONS ELEMENT	67
WEBGROUPS ELEMENT.....	68
VERSION ELEMENT	69
RESULTSET ELEMENT.....	69
SUPPORT.....	70

About Enesys RS Data Extension

Enesys RS Data Extension is a Microsoft SQL Reporting Services Data processing extension that makes it possible to retrieve data from SharePoint lists for the purpose of building reports using Reporting Services.

Once installed, **Enesys RS Data extension** will provide a new type of data source that can be used as part of Reporting Services.

Enesys RS Data Extension provides a specific Query syntax based on Xml that makes it possible to retrieve SharePoint data and apply operations on the data before it is passed to Reporting Services report engine.

License agreement

By installing Enesys RS Data Extension (herein the “Software”) developed by Enesys, you are accepting the following License Agreement. **IMPORTANT:** this license is a legal agreement between you (either an individual or a single entity) and ENESYS. By installing and using the software you are agreeing to be bound by the terms of this license agreement. Read it carefully before installing and using the software. If you do not agree to the terms of this license agreement, then do not install the software.

I. License grants

With respect to the SOFTWARE, the terms of this Agreement supersede and replace any conflicting or contradictory terms contained in any preexisting agreement between Enesys and you.

Enesys grants you a non-exclusive, non-transferable license to install and use the software only as authorized below.

A. Trial License

You are granted a license for evaluation purposes only. You are authorized to install and use the SOFTWARE for the sole purpose of testing its functionalities.

B. Server License

This license grants you the right to install the SOFTWARE on one (and only one) server instance where Microsoft SQL Reporting Services is installed. In a web farm scenario with multiple report server computers connecting to a report server database, you need a valid license for each computer running the Report Server.

The SOFTWARE may be installed on an unlimited number of Business Intelligence Development Studio installations for the purpose of building and testing reports that will be deployed on a Report Server with a valid license.

Reports using Enesys RS Data Extension features and deployed on a Report Server with a valid license can be run by an unlimited number of users employed by you.

C. Enterprise License

This license grants you the right to install the SOFTWARE on an unlimited number of servers owned by you or any subsidiaries if you are representing a legal entity.

The SOFTWARE may be installed on an unlimited number of Business Intelligence Development Studio installations for the purpose of building and testing reports that will be deployed on a Report Server with a valid license.

Reports using Enesys RS Data Extension features and deployed on a Report Server with a valid license can be run by an unlimited number of users employed by you.

II. Software Maintenance and support

The license entitles the purchaser to support and updates of the SOFTWARE for a period of one year following the purchase of the SOFTWARE. Updates and related supplements of the SOFTWARE provided as part of the updates are governed by this license unless otherwise stated.

III. License limitations

You may not disassemble, decompile, reverse engineer, or attempt in any manner to reconstruct or discover any source code of the SOFTWARE.

You may not rent or provide hosting services using the SOFTWARE.

IV. Publicity

You grant Enesys the right to identify you as a user of the SOFTWARE as part of a "customer list" displayed on our web site. At any point, you can submit a written request via email to contact@enesys.fr to have Enesys remove your name.

V. Limited warranty

Enesys warrants that the SOFTWARE will perform substantially in accordance with the accompanying documentation for a period of thirty days from the date of receipt. Neither Enesys nor its suppliers shall be liable to you or any third party for any indirect, special, incidental, punitive, cover or consequential damages (including, but not limited to, damages for the inability to use equipment or access data, loss of business, loss of profits, business interruption or the like), arising out of the use of, or inability to use, the software and based on any theory of liability including breach of contract, breach of warranty, product liability or otherwise. Enesys's total liability to you for actual damages for any cause whatsoever will be limited to the amount paid by you for the SOFTWARE that caused such damage.

Installation and upgrade

System Requirement

Software	Description
Microsoft SQL Server 2005 Reporting Services	<p>Enesys RS Data Extension (ERSDE) must be installed on an installed Reporting Services server in order to run reports built using ERSDE Data source.</p> <p>Please note that ERSDE is not compatible with SQL 2005 Express as custom extension do not work in that specific configuration.</p> <p>Enesys RS Data Extension 2 is compatible with both 32bits and 64bits version of SQL Server 2005.</p>
MOSS 2007 or WSS V3	The purpose of ERSDE is to retrieve data from SharePoint for building and running reports using Reporting Services.
Business Intelligence Development Studio	<p>Reports are built using Microsoft Business Intelligence Development Studio which is a subset of Visual Studio 2005. It can be installed over an existing Visual Studio 2005 installation and will complement it with additional project types including Reporting Services projects.</p> <p>Enesys RS Data Extension must be installed along Business Intelligence Development Studio for building and testing reports using ERSDE features.</p>

Installation

Overview

Enesys RS Data Extension is provided as a zip file which contains the following files and folders:

Software	Description
Enesys.ReportingServices.Ersde.dll	Reporting Services Data Processing extension.
Enesys RS Data Extension Manual.pdf	User manual.
ErsdeSchema folder	Folder containing Enesys RS Data Extension Schema files that can be used for building query using intellisense.
ConfigFiles folder	Folder containing sample configuration files highlighting configuration that must be made to your own configuration file.
ReportSamples folder	Folder containing sample reports demonstrating various use of Enesys RS Data Extension for retrieving data from SharePoint.
	The folder also contains a backed up SharePoint site that

Software

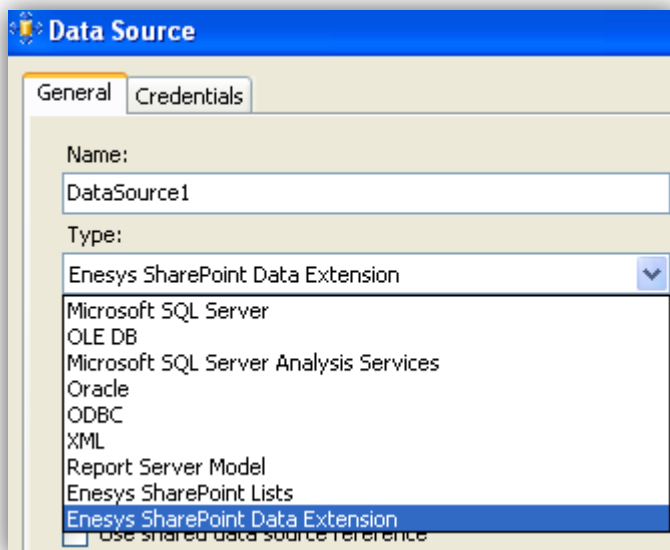
Description

contains the necessary lists for running the sample reports provided.

Though Enesys RS Data Extension must be installed manually, the installation is simple and straightforward.

You will have to install and configure Enesys RS Data Extension both on your Reporting Services server and on any Business Intelligence Development studio installation from which you would like to build and tests reports using ERSDE features.

Once properly installed and configured, **Enesys RS Data Extension** will be available as a data sources type with the name “Enesys SharePoint Data Extension” as shown in the following screenshot:



Report Server installation

Here is a summary of the necessary steps to install and configure Enesys RS Data Extension on a Reporting Services report server.

1

- Copy Enesys RS Data Extension dll on your report server.

2

- Modify **rsreportserver.config** file to let Reporting Services know about the extension.

3

- Modify **rssrvpolicy.config** file to give the necessary permissions to the extension.

Copy Enesys RS Data Extension on the report server

Copy **Enesys.ReportingServices.Ersde.dll** in the bin folder of your Reporting Services server installation.

It will usually be located in the following directory:

[Program files folder]\Microsoft SQL Server\[**Instance folder**] \Reporting Services\ReportServer\bin.

Note:

- Replace **[Program files folder]** by your own program files folder.
- Reporting Services may be installed with any instance of SQL Server. The **[instance folder]** has the following form: MSSQL.X, X being the instance number (e.g.: MSSQL.2, MSSQL.3,...).

Modify rsreportserver.config

Rsreportserver.config file needs to be modified for registering Enesys RS Data Extension as a data processing extension that may be used for running reports.

Rsreportserver.config is located in the following folder:

[Program files folder]\Microsoft SQL Server\[**Instance folder**] \Reporting Services\ReportServer.

1. Open the configuration file.
2. Locate the `<Data>` element.
3. Add the following child node:

```
<Extension Name="ERSDE"  
Type="Enesys.ReportingServices.Ersde.ErsdeConnection,  
Enesys.ReportingServices.Ersde">  
  <Configuration>  
    <LicenseKey></LicenseKey>  
  </Configuration>
```

```
</Extension>
```

4. Save the configuration file.

Note :

- You will need to put your own License key information unless you are installing the evaluation version.
- We recommend that you copy the extension node from the sample **rsreportserver.config** file provided in the package.

Modify rssrvpolicy.config

You need to modify **rssrvpolicy.config** for granting the necessary permissions to the extension.

The **rssrvpolicy.config** file is located in the exact same folder as **rsreportserver.config** file.

1. Open the configuration file.
2. Locate the **CodeGroup** element which **Url** attribute is set to `$(CodeGen)/*` like in the following:

```
<CodeGroup
  class="UnionCodeGroup"
  version="1"
  PermissionSetName="FullTrust">
  <IMembershipCondition
    class="UrlMembershipCondition"
    version="1"
    Url="$(CodeGen)/*"
  />
</CodeGroup>
```

3. Add the following **CodeGroup** element just below the one you have located.

```
<CodeGroup
  class="UnionCodeGroup"
  version="1"
  PermissionSetName="FullTrust">
  <IMembershipCondition
    class="UrlMembershipCondition"
    version="1"
    Url="C:\Program Files\Microsoft SQL Server\MSSQL.2\Reporting
Services\ReportServer\bin\Enesys.ReportingServices.Ersde.dll"
  />
</CodeGroup>
```

4. Save the configuration file.

Note :

- We recommend that you copy the **CodeGroup** node from the sample **rssrvpolicy.config** file provided in the package.
- Whatever the approach you use, you will need to modify the **Url** attribute and enter the path to your Report server bin installation; the full path to the folder where you copied **Enesys.ReportingServices.Ersde.dll**.

Business Intelligence Development studio installation

Like for an installation on the report server, you will need to copy the extension in the appropriate folder and modify two configuration files. All the files are located in the same folder.

Here is a summary of the necessary steps:

- 1** • Copy Enesys RS Data Extension dll in BI development studio PrivateAssemblies folder.
- 2** • Modify RSReportDesigner.config file to let BI development studio know about the extension.
- 3** • Modify RSPreviewPolicy.config file to give the necessary permissions to the extension.

Copy Enesys RS Data Extension

Copy Enesys.ReportingServices.Ersde.dll in the PrivateAssemblies folder of your BI installation which is usually located using the following path:

[Program files folder] \Microsoft Visual Studio 8\Common7\IDE\PrivateAssemblies.

Note:

- Replace **[Program files folder]** by your own program files folder.

Modify RSReportDesigner.config

The **RSReportDesigner.config** file is located in the PrivateAssemblies folder.

1. Open the configuration file.
2. Locate the `<Data>` element.
3. Add the following child node:

```
<Extension Name="ERSDE"
Type="Enesys.ReportingServices.Ersde.ErsdeConnection,
Enesys.ReportingServices.Ersde">
  <Configuration>
    <LicenseKey></LicenseKey>
  </Configuration>
</Extension>
```

4. Locate the designer element.
5. Add the following child node:

```
<Extension Name="ERSDE"
Type="Microsoft.ReportingServices.QueryDesigners.GenericQueryDesigner,
Microsoft.ReportingServices.QueryDesigners"/>
```

6. Save the configuration file

Note :

- You will need to put your own License key information unless you are installing the evaluation version. The same key is used for report server and BI development studio installation.
- We recommend that you copy the extension nodes from the sample **RSReportDesigner.config** file provided in the package.

Modify RSPreviewPolicy.config

The **RSPreviewPolicy.config** file is located in the PrivateAssemblies folder.

1. Open the configuration file.
2. Add the following **CodeGroup** node before the last **CodeGroup** element of the file:

```
<CodeGroup
  class="UnionCodeGroup"
  version="1"
  PermissionSetName="FullTrust">
  <IMembershipCondition
    class="UrlMembershipCondition"
    version="1"
    Url="C:\Program Files\Microsoft Visual Studio
8\Common7\IDE\PrivateAssemblies\Enesys.ReportingServices.Ersde.dll"
  />
</CodeGroup>
```

3. Save the configuration file.

Note :

- We recommend that you copy the **CodeGroup** node from the sample **RSPreviewPolicy.config** file provided in the package.
- Whatever the approach you use, you will need to modify the **Url** attribute and enter the path to your BI development studio PrivateAssemblies folder if it is different from what is shown in this documentation.

Sample configuration files

Enesys RS Data Extension package includes a folder named ConfigFiles which contains sample configuration files for both Reporting Services server and Business Intelligence Development Studio.

Those files are not intended to replace your own configuration files. Their purpose is to highlight the necessary configuration modifications and to let your copy those modification within your own configuration files.

To make it easier, the elements that must be added into your own configuration files have been surrounded by XML comments as in the following example:

```
<Extension Name="XML" Type="Microsoft.ReportingServices
<!-- Enesys RS Data Extension - BEGIN -->
  <Extension Name="ERSDE" Type="Enesys.ReportingServices
    <Configuration>
      <LicenseKey>|</LicenseKey>
    </Configuration>
  </Extension>
<!-- Enesys RS Data Extension - END -->
</Data>
<SemanticQuery>
```

We recommend that you copy the various elements from those sample configuration files rather than from the documentation.

Installing schema files

If you would like to benefit from IntelliSense when writing an Enesys RS Data Extension query, you can install the schema files provided with the package in the following folder:

```
%Program Files%\Microsoft Visual Studio 8\Xml\Schemas
```

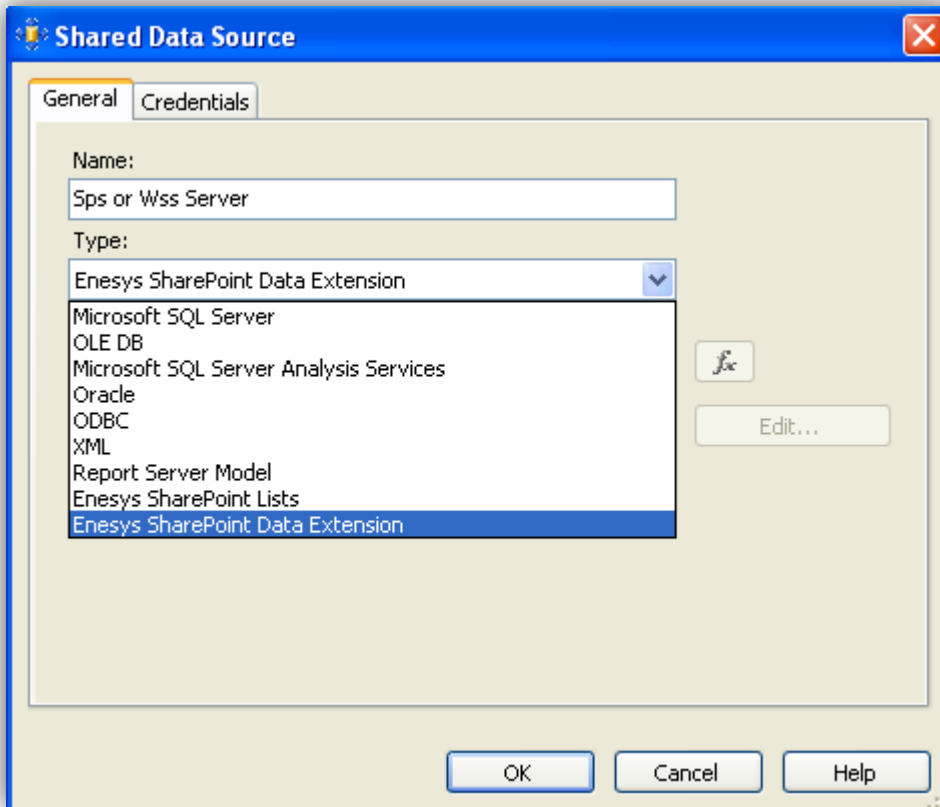
For more information on using intellisense see “Using Intellisense for writing queries” page “55”.

Upgrading from version 1.x

Overview

Enesys RS Data Extension version 2.x is not compatible anymore with SharePoint 2003 and Windows SharePoint Services 2. Support for SQL Server 2000 has also been discontinued in this version and the extension will only install on SQL Server Reporting Services 2005.

The extension has been renamed for making it possible to install it side by side with a previous version. The name that will display when selecting the type of data source has also been changed to “**Enesys SharePoint Data Extension**” compared to “**Enesys SharePoint Lists**” for the previous versions as shown in the following screenshot:



Even though you would not have to retrieve data from both SharePoint 2003 and SharePoint 2007, you can still install the new version side by side with the previous version if you would like to make some tests first.

Upgrading

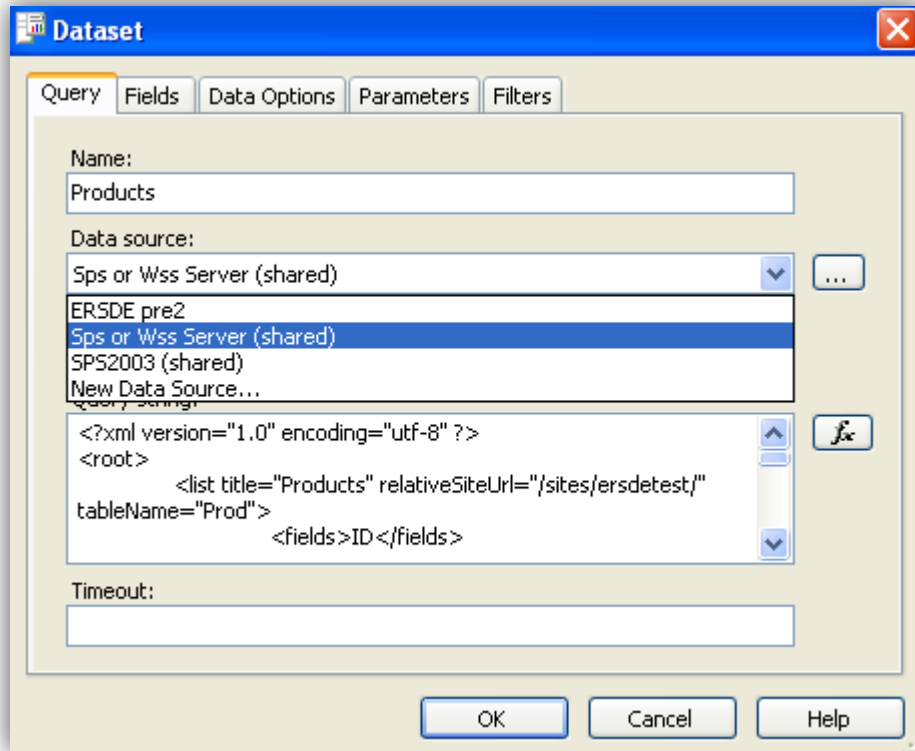
Just follow the installation instructions provided in this documentation.

Once you have installed the new version, you will need to do one of the following in order to run reports using **Enesys RS Data Extension 2.x**.

Gradual approach

Create a new shared data source of type “Enesys SharePoint Data Extension”.

On a report by report basis, change the data source of the report’s dataset so that it points to the data source using the new version of **Enesys RS Data Extension** as shown in the following screenshot:



Notes:

A report may have several datasets defined (especially if you are setting report parameter available values using a dataset). Though it is possible on a technical standpoint to have several datasets using different data sources, it is not recommended to use two different version of **Enesys RS Data Extension** in a same report.

Straight approach

If you would like to use the new version of Enesys RS Data Extension with all of your reports instead of on a report by report basis, you will only need to change the type of your shared data source from “Enesys SharePoint Lists” to “Enesys SharePoint Data Extension”.

Complementary notes:

Once you have made your modification to either data sources or datasets, do not forget to re deploy your shared data sources and/or reports to your server. Alternatively, you may make the modification directly on the report server using the web management interface.

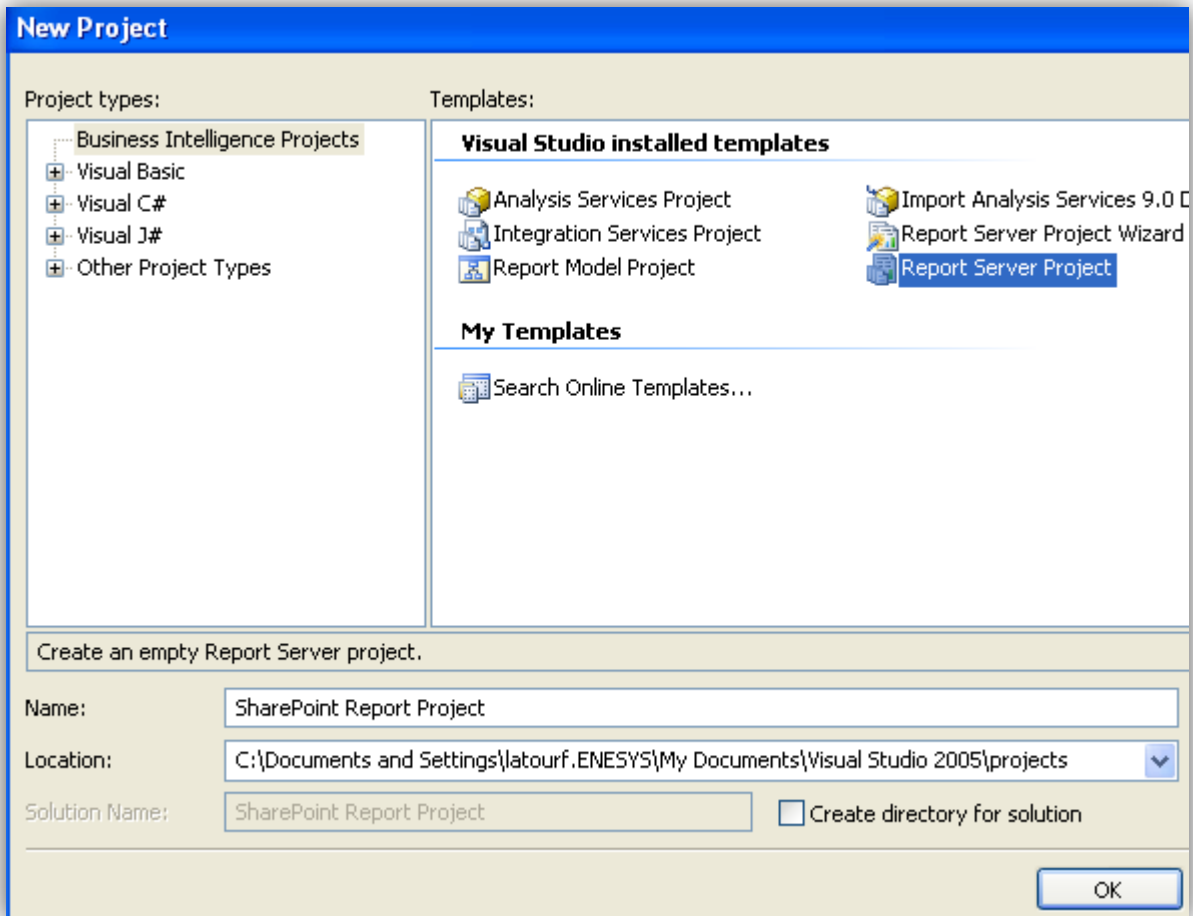
If you have data sources definitions embedded within your reports, you will not have any other alternative than modifying your reports in order to use **Enesys RS Data Extension 2.x**.

Quick start

This chapter shows how to quickly create a report using data from a SharePoint list. If you have already built reports with Business Intelligence Development Studio, you can skip this section and go to "Using Enesys RS Data Extension" section.

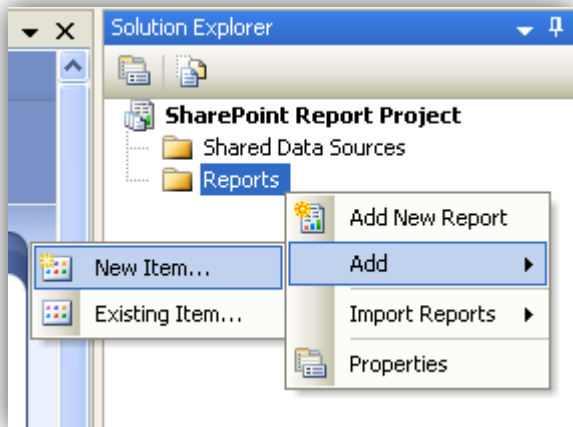
Creating a new report project

Open Business Intelligence Development Studio or Visual Studio 2005 and create a new Report Server Project.

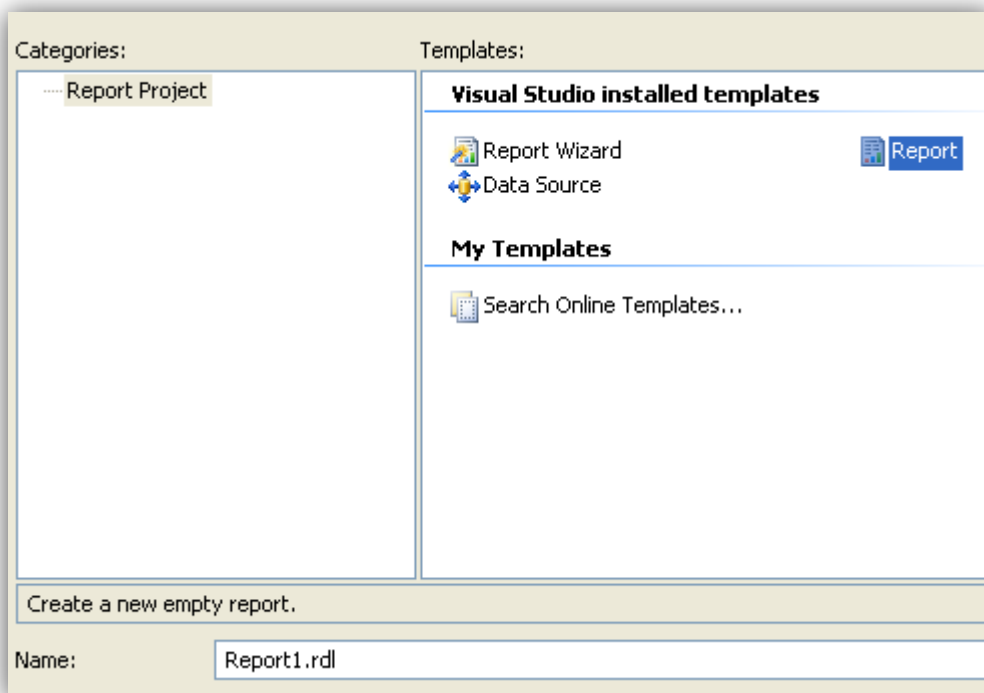


Adding a new report

Right-click on Reports in the solution explorer and add a new item:



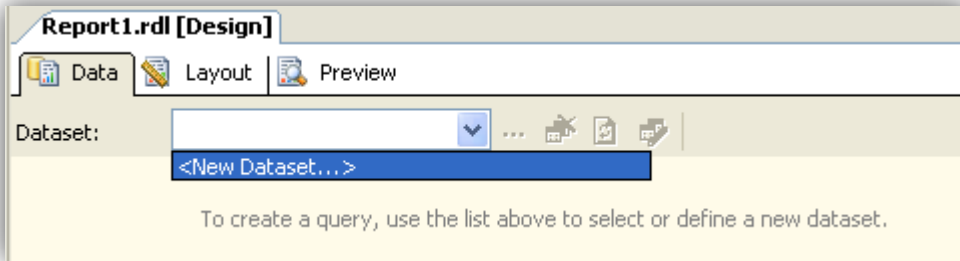
From the list of templates displayed, select "Report" and click on the **Add** button:



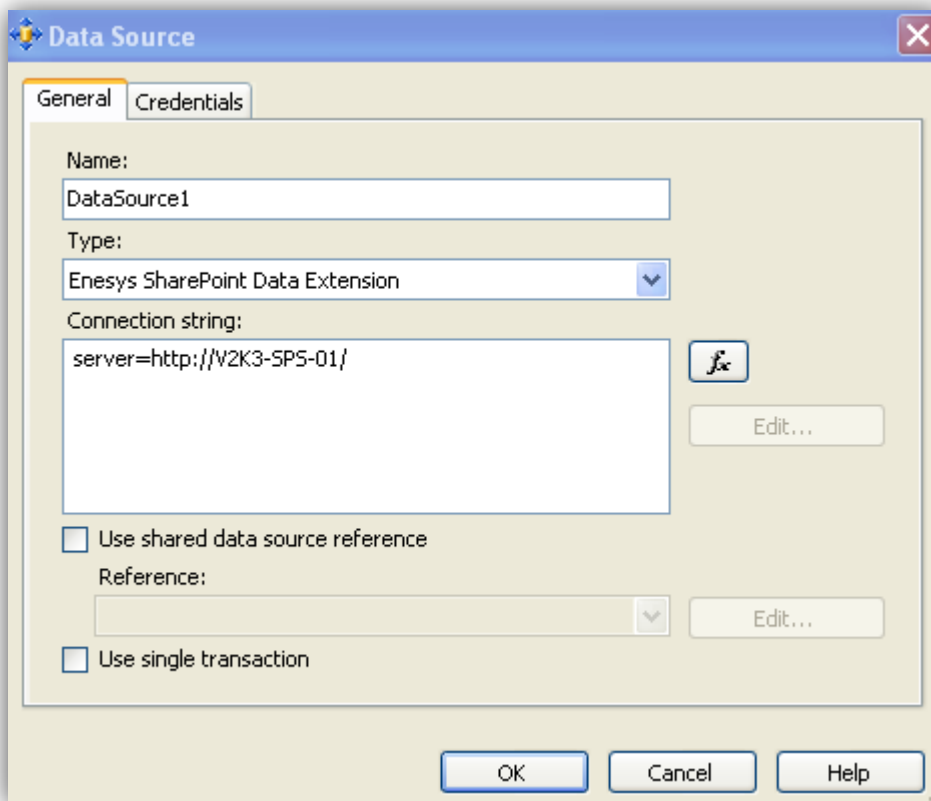
A new empty report named Report1.rdl will be created.

Creating a dataset

Develop the **Dataset** scroll-down list and select **New Dataset**.



As no shared data source is defined, you will be proposed to create a data source:



Enter the following information:

Field	Description
Name	Name that you would like to give to your data source; for example, the name of the SharePoint server from which you will retrieve data.
Type	Select "Enesys SharePoint Data Extension" from the scroll-down list.
Connection string	Enter "server=SharePoint Url". Replace SharePoint Url with the URL of your SharePoint server (e.g. http://intranet).

Select the **Credentials** tab and use the desired authentication method. Windows Authentication should be fine for the purpose of testing a simple report provided that you have the necessary rights to read SharePoint lists.


Click **OK** to create the dataset. Now you must create the query that will make it possible to retrieve data from SharePoint lists.

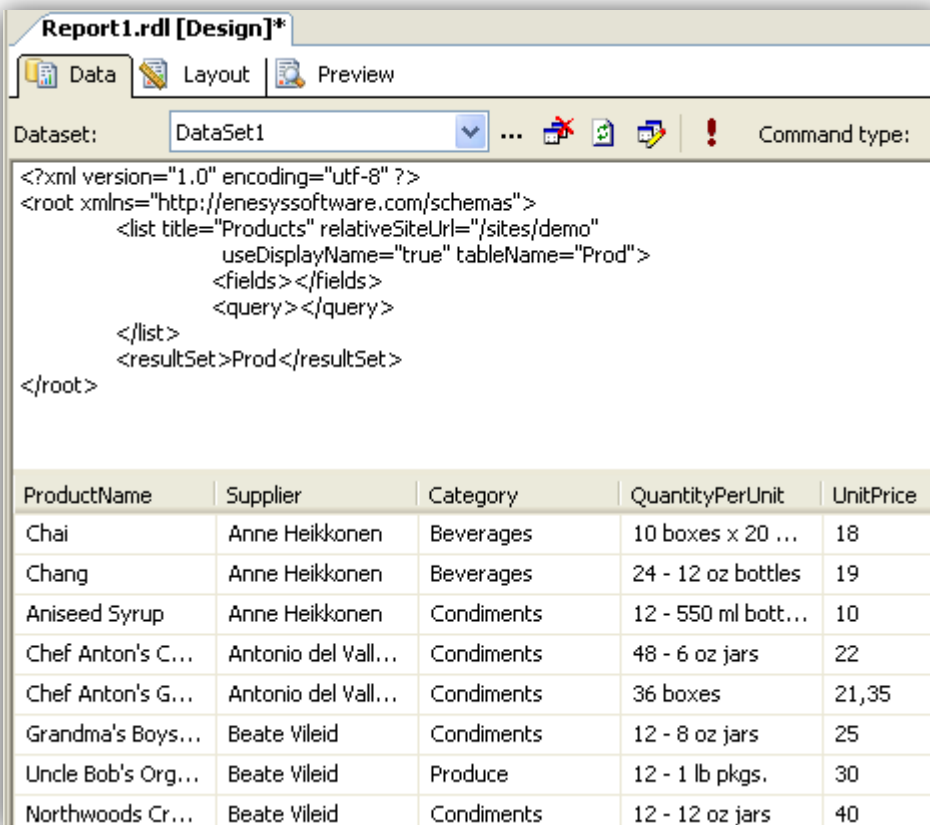
Creating a query

Write a very simple query that will return data from one of your SharePoint list as in the following:

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <list title="Products" relativeSiteUrl="/sites/demo"
        useDisplayName="true" tableName="Prod">
    <fields></fields>
    <query></query>
  </list>
  <resultSet>Prod</resultSet>
</root>
```

Replace “Products” and “/sites/demo/” with the names of a list and a SharePoint site located on your server.

Click on the run  button to execute the query. You should get all the items from your SharePoint list:




The screenshot shows the Report Designer interface for 'Report1.rdl [Design]*'. The 'Data' tab is active, showing a dataset named 'DataSet1'. The XML query is displayed in the main area, and the resulting data is shown in a table below.

ProductName	Supplier	Category	QuantityPerUnit	UnitPrice
Chai	Anne Heikkonen	Beverages	10 boxes x 20 ...	18
Chang	Anne Heikkonen	Beverages	24 - 12 oz bottles	19
Aniseed Syrup	Anne Heikkonen	Condiments	12 - 550 ml bott...	10
Chef Anton's C...	Antonio del Vall...	Condiments	48 - 6 oz jars	22
Chef Anton's G...	Antonio del Vall...	Condiments	36 boxes	21,35
Grandma's Boys...	Beate Vileid	Condiments	12 - 8 oz jars	25
Uncle Bob's Org...	Beate Vileid	Produce	12 - 1 lb pkgs.	30
Northwoods Cr...	Beate Vileid	Condiments	12 - 12 oz jars	40

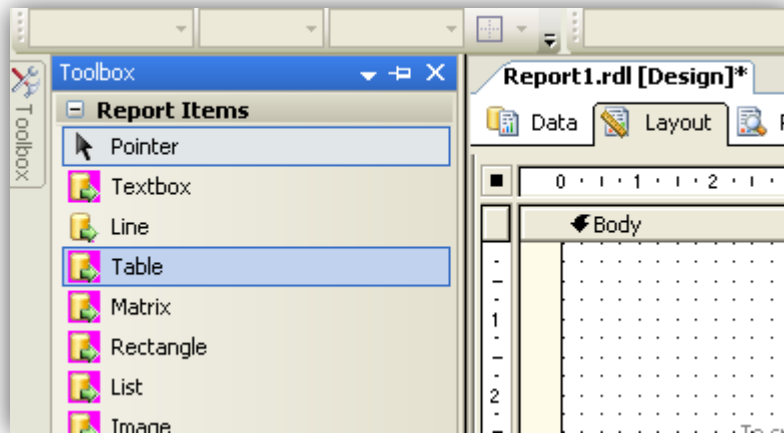
Once the query has been tested, the logical approach is, of course, to design a layout for the data.

Layout

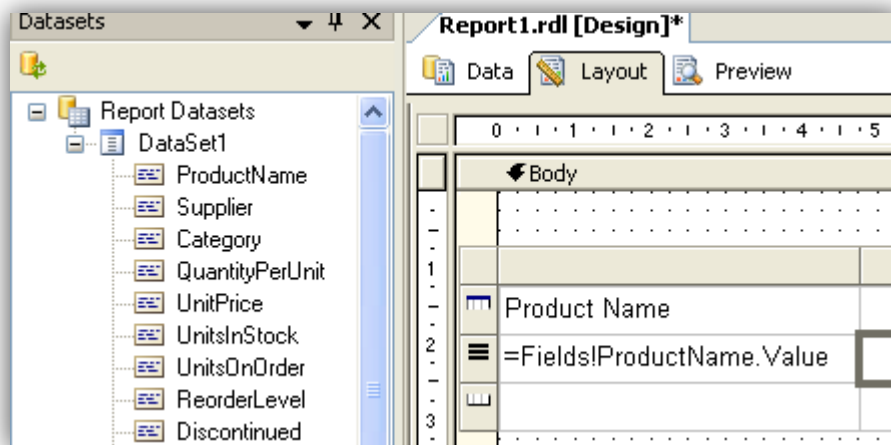
Note that if you have just created the dataset, you may have to click the “refresh fields”  button to display the list of fields that can be used in the formatting.

Select the Layout view.

Drag the **Table** component from the toolbox available at the left side of the report designer



Drag the desired fields from the SharePoint list into the table you have just added to the layout:



Change the style of the various cells to make the report more visually appealing and select Preview to see the result.

Conclusion

For the most part, the information presented in this quick start guide is not specific to **Enesys RS Data Extension**. The approach is similar regardless of the type of data source used.

What you should retain is:

- The data source must be of the type Enesys SharePoint Data Extension.

- The connection string for the data source must be in the form of "server=<Url>" where <Url> corresponds to the URL of the SharePoint server which data is being used.
- The dataset used to retrieve SharePoint data is built using a specific query string in XML format. The main purpose of this documentation is to explain the details of the specific query syntax.

Using Enesys RS Data Extension

Retrieving data from a SharePoint list

You can retrieve data from any SharePoint list using **Enesys RS Data Extension** specific query syntax which is based on XML.

In its simplest form, a query string is written as follows:

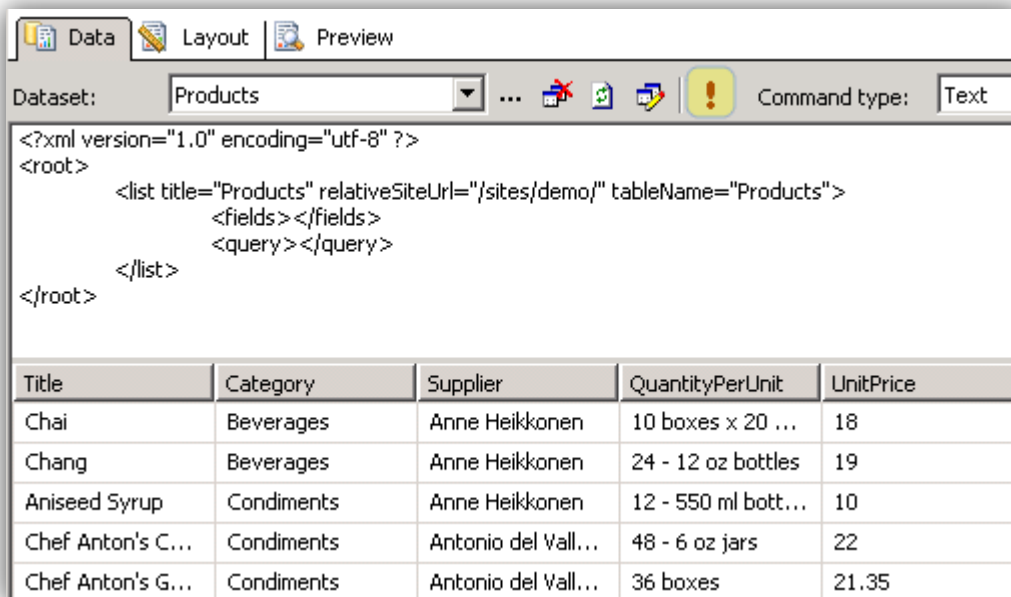
```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" tableName="Products">
    <fields></fields>
    <query></query>
  </list>
</root>
```

The **list** element makes it possible to obtain data from a SharePoint list.

The attributes and the child elements make it possible to specify the desired list, as well as any selection criteria.

The previous query retrieves all the items from the SharePoint list “**Products**” located on the site “</sites/demo/>”.

Granted that you have created a data source of type “Enesys SharePoint Data Extension”, this is all there needs to be to run the query within the report designer and retrieve the list items:



The screenshot shows the Enesys RS Data Extension report designer interface. The 'Dataset' dropdown is set to 'Products' and the 'Command type' is 'Text'. The XML query is displayed in the main area, and the results are shown in a table below.

Title	Category	Supplier	QuantityPerUnit	UnitPrice
Chai	Beverages	Anne Heikkonen	10 boxes x 20 ...	18
Chang	Beverages	Anne Heikkonen	24 - 12 oz bottles	19
Aniseed Syrup	Condiments	Anne Heikkonen	12 - 550 ml bott...	10
Chef Anton's C...	Condiments	Antonio del Vall...	48 - 6 oz jars	22
Chef Anton's G...	Condiments	Antonio del Vall...	36 boxes	21.35

All columns are returned because the **fields** element has been left empty.

All items are returned because no criterion was specified in the **query** element.

Specifying SharePoint list

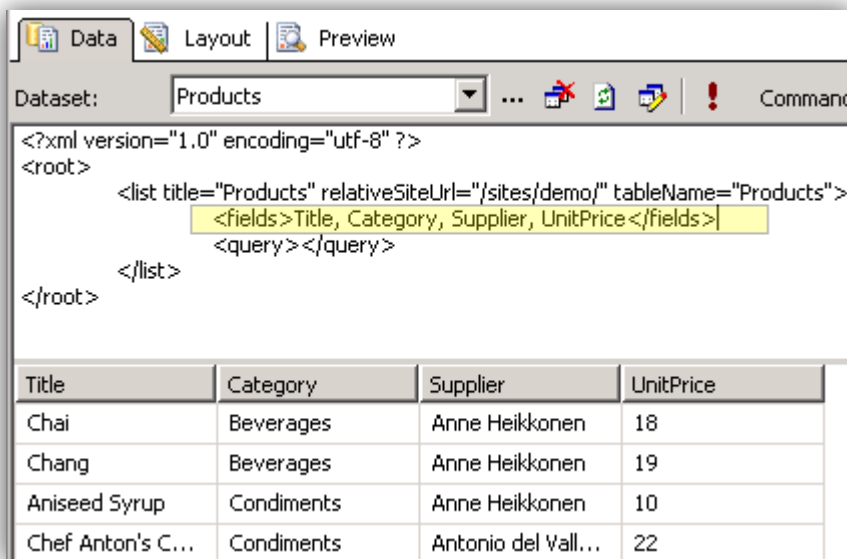
A SharePoint list title can easily be modified by a user. To avoid this situation you may specify the SharePoint list to retrieve items from, by specifying its ID (also called name) using the **listID** attribute rather than specifying its title using **title** attribute.

The SharePoint list ID will not change over time. It still can be deleted.

Note that if you are using both attributes, **listID** will take the precedence over **title**. Even if you rely on list id to specify a SharePoint list, we encourage you to set title attribute as a meaningful reminder of list content.

Specifying list columns

If you don't need all the columns of the SharePoint list, you can specify the columns you would like to retrieve using the **fields** element:



The screenshot shows a software interface with a 'Dataset:' dropdown set to 'Products'. Below it, a text area contains the following XML query:

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" tableName="Products">
    <fields>Title, Category, Supplier, UnitPrice</fields>
    <query></query>
  </list>
</root>
```

Below the XML, a table displays the data retrieved from the list:

Title	Category	Supplier	UnitPrice
Chai	Beverages	Anne Heikkonen	18
Chang	Beverages	Anne Heikkonen	19
Aniseed Syrup	Condiments	Anne Heikkonen	10
Chef Anton's C...	Condiments	Antonio del Vall...	22

Filtering list data

A filter can be applied to a list using Collaboration Application Markup Language query format. It doesn't take a long time to understand the basic principles. The CAML Query must be placed within the **<query>** element. The following query example will retrieve items where the **Category** column equals to "Condiments":

The screenshot shows a reporting tool interface with a 'Dataset' dropdown set to 'Products'. The 'Command type' is set to 'Query'. The XML query is as follows:

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" tableName="Products">
    <fields>Title, Category, Supplier, UnitPrice</fields>
    <query>
      <Where>
        <Eq>
          <FieldRef Name="Category" />
          <Value Type="Text">Condiments</Value>
        </Eq>
      </Where>
    </query>
  </list>
</root>
```

Below the query, a table displays the filtered results:

Title	Category	Supplier	UnitPrice
Aniseed Syrup	Condiments	Anne Heikkonen	10
Chef Anton's C...	Condiments	Antonio del Vall...	22
Chef Anton's G...	Condiments	Antonio del Vall...	21.35

You can also apply a filter at the report level using reporting services features. However, in that case, the whole list data is retrieved before the filter is applied. CAML filter is applied at SharePoint server side and it will improve the performance considerably if you just need a subset of the list. You can also use a mix of those filtering options when necessary.

Using column display names or internal names

SharePoint list columns have an internal name and a display name. When a column is initially created, the display name and the internal name of the column are the same (except if the name contains space or accent marks). When you modify the name of a column, it will only modify the display name. The internal name is never modified and you may end up with columns whose internal names no longer have any connections to the display names.

Though using internal names has the advantage of not breaking your report when a column name is changed, you can choose to use display names when specifying columns to retrieve (fields element) and filtering the list (query element) by setting the `useDisplayName` attribute to true.

The screenshot shows a report designer window with the following XML code:

```

<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" useDisplayName="true" tableName="Products">
    <fields>ProductName, Category, Supplier, UnitPrice</fields>
    <query>
      <Where>
        <Contains>
          <FieldRef Name="ProductName" />
          <Value Type="Text">Hot</Value>
        </Contains>
      </Where>
    </query>
  </list>
</root>

```

Below the XML code is a table with the following data:

ProductName	Category	Supplier	UnitPrice
Louisiana Fiery Hot Pepper S...	Condiments	Antonio del Vall...	21.05
Louisiana Hot Spiced Okra	Condiments	Antonio del Vall...	17

Using report parameters

Reporting Services lets you define parameters at the report level so that the user may be proposed several options for running the report.

Parameters may be used within the **query** element. A parameter is composed of a name surrounded by the characters @ and ! (e.g. @product!).

When you use a parameter in the query string and it is not defined at the report level, it will be automatically created when you select the Layout tab.

Also note that the query execution in the data window will ask you to enter a value for the parameter.

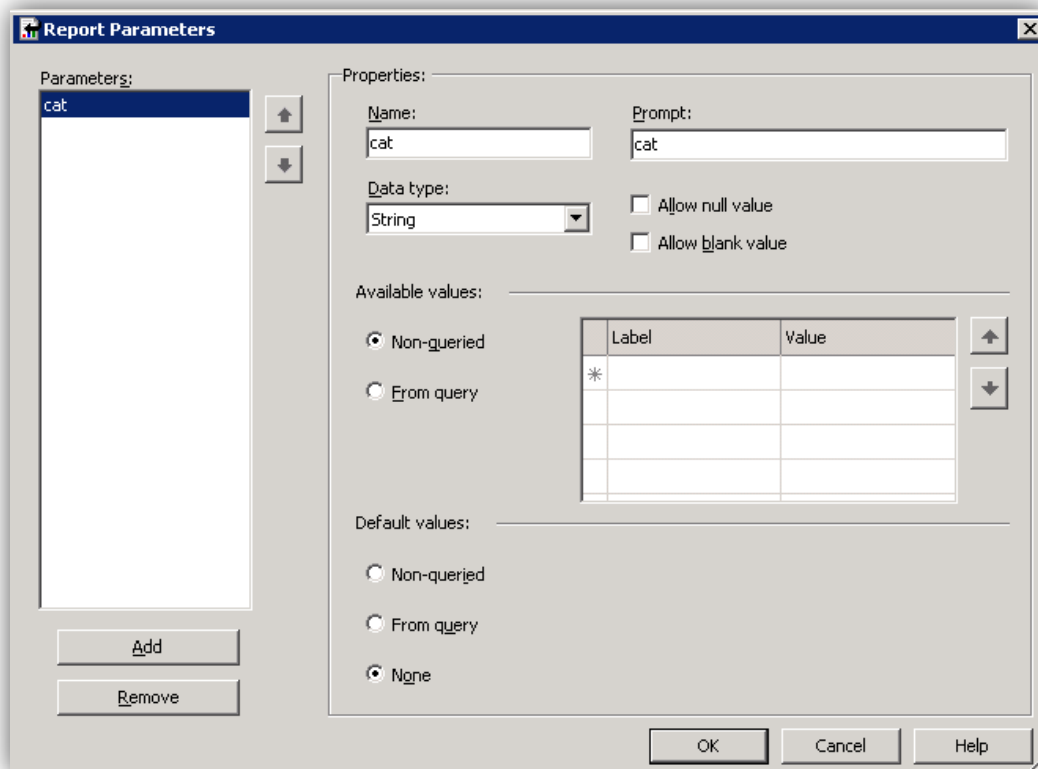
The following example shows how to return the data from the SharePoint "Products" list whose category is equal to the value of the "cat" parameter. The parameter's value will be entered when the report is executed and inserted into the query string.

```

<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" tableName="Products">
    <fields>Title, Category, Supplier, UnitPrice</fields>
    <query>
      <Where>
        <Eq>
          <FieldRef Name="Category" />
          <Value Type="Text">@cat!</Value>
        </Eq>
      </Where>
    </query>
  </list>
</root>

```

As previously indicated, entering the parameter in the query string will automatically create the corresponding report parameter, as shown in the following image:



Rather than letting the category be entered in full text, you can create a new dataset that will be used to define the available values, from which the report user shall make a selection.

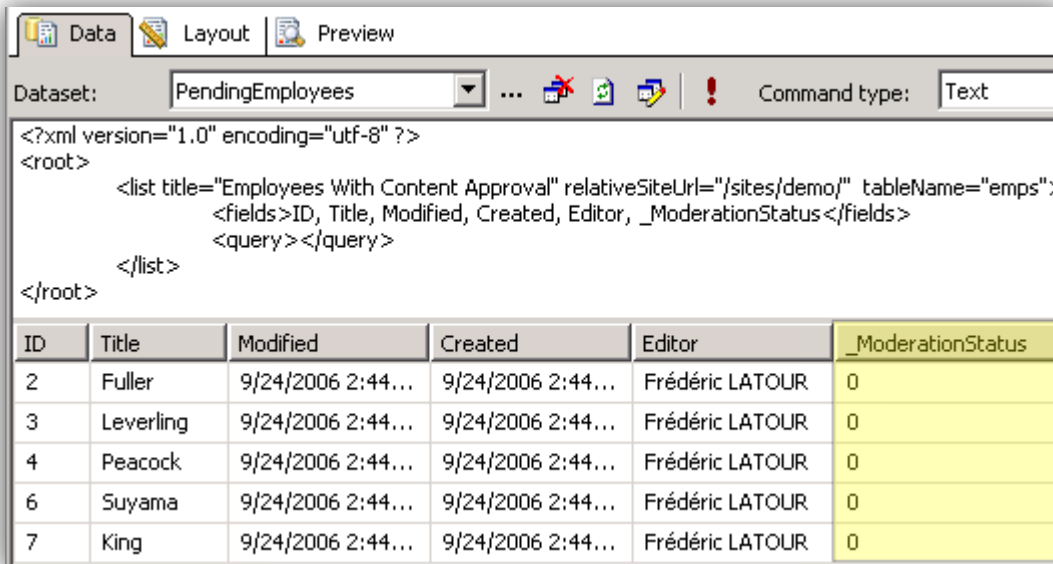
If you have a distinct SharePoint list containing the list of categories, you can simply create a dataset returning the list data.

To create a dataset from the list of "products" while eliminating repeats (a category may be associated with several products and thus appear several times in the list), you must apply a "distinct" operation to the list as it will be explained later.

Note that you can use any other data source (e.g. SQL) for the available values.

Pending items

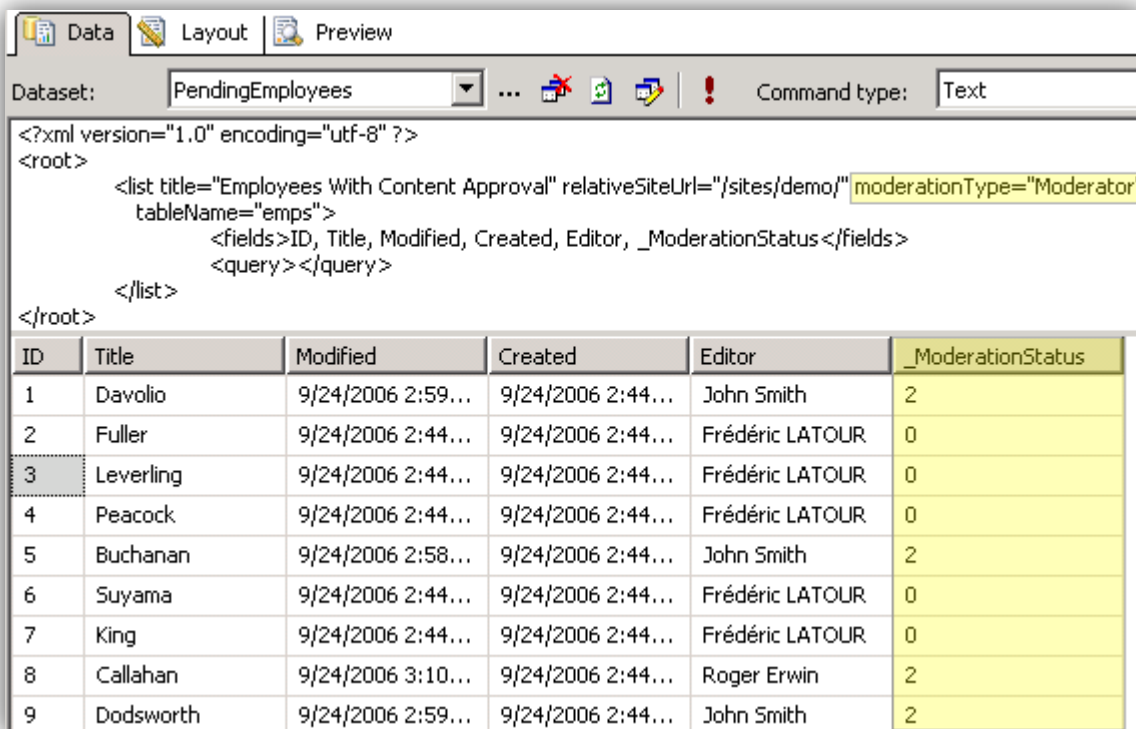
For lists for which content approval is required, only approved items are retrieved by default as shown by the **_ModerationStatus** column equals to 0:



The screenshot shows a SharePoint list view titled "Employees With Content Approval". The list contains 7 items, all with a _ModerationStatus of 0. The XML view shows the list definition with the following fields: ID, Title, Modified, Created, Editor, and _ModerationStatus.

ID	Title	Modified	Created	Editor	_ModerationStatus
2	Fuller	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
3	Leverling	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
4	Peacock	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
6	Suyama	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
7	King	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0

You can use the **moderationType** attribute to retrieve pending items for a specific list:



The screenshot shows a SharePoint list view titled "Employees With Content Approval" with the **moderationType="Moderator"** attribute. The list contains 9 items with various _ModerationStatus values. The XML view shows the list definition with the following fields: ID, Title, Modified, Created, Editor, and _ModerationStatus.

ID	Title	Modified	Created	Editor	_ModerationStatus
1	Davolio	9/24/2006 2:59...	9/24/2006 2:44...	John Smith	2
2	Fuller	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
3	Leverling	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
4	Peacock	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
5	Buchanan	9/24/2006 2:58...	9/24/2006 2:44...	John Smith	2
6	Suyama	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
7	King	9/24/2006 2:44...	9/24/2006 2:44...	Frédéric LATOUR	0
8	Callahan	9/24/2006 3:10...	9/24/2006 2:44...	Roger Erwin	2
9	Dodsworth	9/24/2006 2:59...	9/24/2006 2:44...	John Smith	2

Expanding recurring events

The **expandRecurrent** attribute let's you expand recurring events for an event list.

You will appreciate this feature as even SharePoint object model does not include this possibility. Besides building reports displaying recurring events, a possible scenario is to export

recurring events as an xml file using Reporting Services subscription features in order to feed another data source or a business process.

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Events" relativeSiteUrl="/sites/demo/" expandRecurrent="true"
    expandFirstDate="@firstDate!" expandLastDate="@lastDate!"
    tableName="Events">
    <fields></fields>
    <query>
    </query>
  </list>
</root>
```

The `expandRecurrent` attribute goes along with `expandFirstDate` and `expandLastDate` optional attributes that lets you define the range of dates for which recurring events will be expanded. Parameters may be used to set `expandFirstDate` and `expandLastDate` values.

It is important to note that the CAML Query will not filter the expanded events. Recurring events are expanded after they have been retrieved from the SharePoint list.

Let's see an example to clarify this point:

```
<root>
  <list title="Events" relativeSiteUrl="/sites/demo/" expandRecurrent="true"
    tableName="Events">
    <fields></fields>
    <query>
      <Where>
        <Geq>
          <FieldRef Name="EventDate" />
          <Value Type="DateTime">@StartDate!</Value>
        </Geq>
      </Where>
    </query>
  </list>
</root>
```

One may think that the previous query would retrieve all events (including recurring events) starting from "StartDate" parameter. This is not exactly the case. Recurring events starting before "StartDate" parameter won't be retrieved at all though after being expanded some events may indeed start after "StartDate" parameter. Only recurring events starting from "StartDate" will be expanded.

To make the query as easy to write would have made it necessary to write our own caml interpreter in order to filter the expanded events with the relevant part of the caml query.

Nevertheless, there is a solution to obtain events between two dates including the recurring events by writing the following query:

```

<root>
  <list title="Events" relativeSiteUrl="/sites/demo/" expandRecurrent="true"
    expandFirstDate="@firstDate!" expandLastDate="@lastDate!"
    tableName="RecEvents">
    <fields></fields>
    <query>
      <Where>
        <And>
          <And>
            <Leq>
              <FieldRef Name="EventDate" />
              <Value Type="DateTime">@lastDate!</Value>
            </Leq>
            <Geq>
              <FieldRef Name="EndDate" />
              <Value Type="DateTime">@firstDate!</Value>
            </Geq>
          </And>
          <Eq>
            <FieldRef Name="fRecurrence" />
            <Value Type="Boolean">1</Value>
          </Eq>
        </And>
      </Where>
    </query>
  </list>

```

```

<list title="Events" relativeSiteUrl="/sites/demo/" expandRecurrent="false"
  tableName="Events">
  <fields></fields>
  <query>
    <Where>
      <And>
        <And>
          <Geq>
            <FieldRef Name="EventDate" />
            <Value Type="DateTime">@firstDate!</Value>
          </Geq>
          <Leq>
            <FieldRef Name="EventDate" />
            <Value Type="DateTime">@lastDate!</Value>
          </Leq>
        </And>
        <Eq>
          <FieldRef Name="fRecurrence" />
          <Value Type="Boolean">0</Value>
        </Eq>
      </And>
    </Where>
  </query>
</list>

```

```

<sqlOp op="union">
  <dstTableName>MergedEvents</dstTableName>
  <parentTableName>RecEvents</parentTableName>
  <childTableName>Events</childTableName>
  <labelColumn>>true</labelColumn>
  <parentLabelValue>Recurrent</parentLabelValue>
  <childLabelValue>Non Recurrent</childLabelValue>
</sqlOp>

<resultSet>MergedEvents</resultSet>
</root>

```

The query has been separated in three parts for clarity. You can copy it from the sample report “Recurring Events Real World” provided as part of the package.

The first **list** query element will retrieve recurring items only (fRecurrence) and expand them using the date range defined by “firstDate” and “lastDate” parameters.

The second **list** query element will retrieve non recurring events where EventDate is between “firstDate” and “lastDate” parameters.

At last, **sqlOp op="union"** operation will merge both result sets into one dataset suitable for Reporting Services.

Stripping Html Tags

It is a well known limitation that Reporting Services is not able to handle html tags within a specific field. Thus, html data within a field will be displayed as plain text as shown in the following image:

Title	Description
Html test	<pre><div>Bold text</div> <div>Italic text</div> <div align=center>Reporting Services does not support html data.</div> <div align=center>It will display html tags as plain text.</div> <div>&nbsp;</div></pre>
Essential Resources for SharePoint Developers	<pre><p>Some interesting resources for SharePoint Developers</p> Microsoft Office SharePoint Server (MOSS) SDK and ECM Starter Kit Windows SharePoint Services (WSS) SDK and Workflow Starter Kit Visual Studio Extensions for SharePoint Services (November CTP) Customizing and Branding Web Content Management-Enabled SharePoint Sites MOSS for Content Management Server Developers (Beta) Office Developer Screencasts (applies to all of Office) SharePoint Developer Map (also includes InfoPath and 2007 Office System posters) MOSS and WSS Online Clinics MOSS portal on the Office Developer Center SharePoint Developer Center 7 Development Projects for SharePoint – online book MSDN Community Content (WSS & MOSS) F1 Help from Visual Studio Document Explorer Project SDK Download
</pre>

Though, you may use your own approach to remove html tags by using Reporting Services embedded code features, we have added the ability to strip html tags for a specific SharePoint list using the `stripHtml` attribute:

```
<root>
  <list title="Html Sample" relativeSiteUrl="/sites/demo/" stripHtml="true"
    tableName="html">
    <fields></fields>
    <query>
    </query>
  </list>

  <resultSet>html</resultSet>
</root>
```

The `stripHtml` attribute is optional and defaults to false if it's not defined.

Though not really appealing, stripping html makes text at least readable:

Title	Description
Html test	<p>Bold text</p> <p>Italic text</p> <p>Reporting Services does not support html data. It will display html tags as plain text.</p>
Essential Resources for SharePoint Developers	<p>Some interesting resources for SharePoint Developers</p> <ul style="list-style-type: none"> - Microsoft Office SharePoint Server (MOSS) SDK and ECM Starter Kit - Windows SharePoint Services (WSS) SDK and Workflow Starter Kit - Visual Studio Extensions for SharePoint Services (November CTP) - Customizing and Branding Web Content Management-Enabled SharePoint Sites - MOSS for Content Management Server Developers (Beta) - Office Developer Screencasts (applies to all of Office) - SharePoint Developer Map (also includes InfoPath and 2007 Office System posters) - MOSS and WSS Online Clinics - MOSS portal on the Office Developer Center - SharePoint Developer Center - 7 Development Projects for SharePoint – online book - MSDN Community Content (WSS & MOSS) - F1 Help from Visual Studio Document Explorer - Project SDK Download

Using Running Values

Though Reporting Services makes it possible to calculate running values on the fly, it will not let you use aggregate functions on running values (max, min, etc).

To overcome this limitation and being able to meet specific scenarios, Enesys RS Data Extension lets you specify columns for handling running values. In that case, the running value being seen as a column on Reporting Services side, it is possible to apply aggregation function on it.

Columns containing running values are added using a `<customFields>` child element of the `<list>` element as shown in the following example:


```

<root>
  <list title="Order Details" relativeSiteUrl="/sites/demo/" tableName="Order Details">
    <fields>Product, Quantity</fields>
    <query>
      <OrderBy>
        <FieldRef Name="Product" />
      </OrderBy>
    </query>
    <customFields>
      <field name="RunningTotal" dataType="System.Int32" op="RunningSum"
        groupColumnName="Product" param="Quantity"></field>
    </customFields>
  </list>

  <resultSet>Order Details</resultSet>
</root>

```

Each <field> child element of <customFields> element represents a specific column holding a running value. The attributes let's you specify the name of the new column that will be holding the running value, the data type of the column, the type of running value, the grouping column and the column holding the value.

To make it simple, the previous query will retrieve the Product and Quantity columns from the "Order Details" SharePoint list, ordered by "Product". Enesys RS Data Extension will add a custom column named "RunningTotal" (**name**) of type Integer (**dataType**) and will calculate a running sum (**op**) based on the "Quantity" column (**param**). Each time the "Product" column value changes (**groupColumnName**), the running sum is reset to 0.

The following image is what you will obtain when running the previous query within the Data view.

Note that the Running Sum is reset when the product value changes from "Aniseed Syrup" to "Boston Crab Meat".

It is important to note also that Enesys RS Data Extension will not automatically order the data on the **groupColumnName** column ("Product" in that case). It's up to you to order accordingly the SharePoint list using an <OrderBy> element within the CAML query. The **groupColumnName** attribute will only direct ERSDE to reset the running value when the **groupColumnName**'s column value changes.

```

<root>
  <list title="Order Details" relativeSiteUrl="/sites/demo/" tableName="Order Details">
    <fields>Product, Quantity</fields>
    <query>
      <OrderBy>
        | <FieldRef Name="Product" />
      </OrderBy>
    </query>
    <customFields>
      <field name="RunningTotal" dataType="System.Int32" op="RunningSum"
        groupColumnName="Product" param="Quantity"></field>
    </customFields>
  </list>
  <resultSet>Order Details</resultSet>
</root>

```

Quantity	Product	RunningTotal
6	Aniseed Syrup	180
20	Aniseed Syrup	200
20	Aniseed Syrup	220
49	Aniseed Syrup	269
30	Aniseed Syrup	299
25	Aniseed Syrup	324
4	Aniseed Syrup	328
50	Boston Crab Meat	50
60	Boston Crab Meat	110

Retrieving items and folders in a specific folder

The default behaviour when retrieving items using a `list` element is to returned all items within all folders and subfolders.

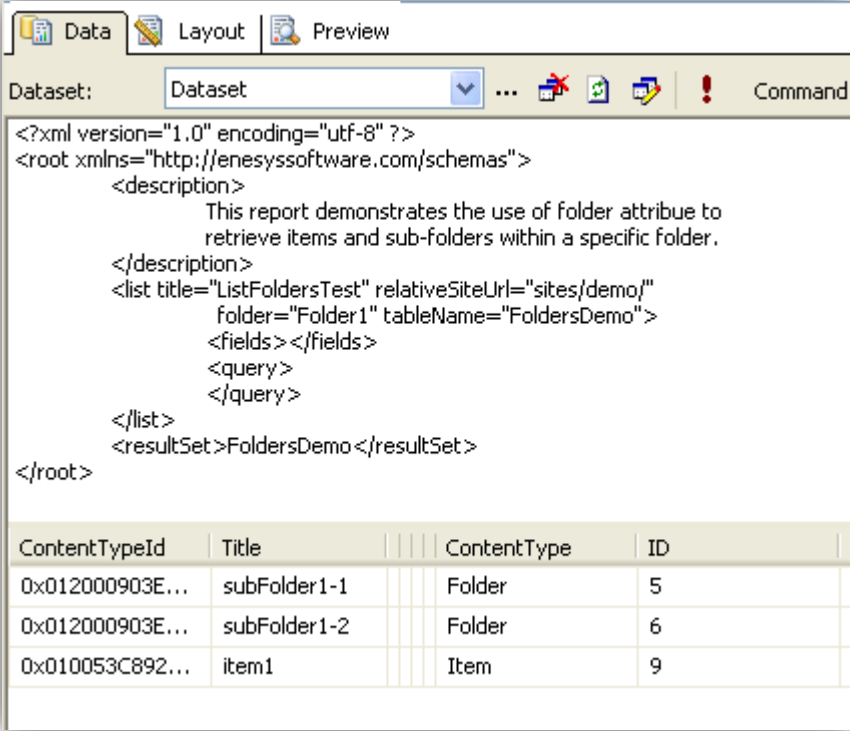
The optional `folder` attribute lets you retrieve items and folders in a specific SharePoint list folder as shown in the following query:

```

<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This report demonstrates the use of folder attribute to
    retrieve items and sub-folders within a specific folder.
  </description>
  <list title="ListFoldersTest" relativeSiteUrl="sites/demo/"
    folder="Folder1" tableName="FoldersDemo">
    <fields></fields>
    <query>
    </query>
  </list>
  <resultSet>FoldersDemo</resultSet>
</root>

```

When specifying a folder attribute, you will retrieve items located in the specified folder as well as subfolder as shown in the following screenshot:



To retrieve root items and folders, set the **folder** attribute to a "/" (slash) value.

Instead of specifying the folder using a literal value, you may use a report parameter (e.g.: **folder="@someReportParameter!"**).

Retrieving the first n items of a list

The **rowLimit** attribute let's you specify the number of items to retrieve.

The following query will retrieve only the first five less expensive products:

```

<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This report demonstrates the use of RowLimit attribute
    to retrieve only the top n items of the list.
  </description>
  <list title="Products" relativeSiteUrl="/sites/demo/"
    rowLimit="5" tableName="Products">
    <fields>Title, Supplier, Category, UnitPrice</fields>
    <query>
      <OrderBy>
        <FieldRef Name="UnitPrice"></FieldRef>
      </OrderBy>
    </query>
  </list>
  <resultSet>Products</resultSet>
</root>

```

Custom separator multiple values columns

The following type of SharePoint columns may have multiple values:

- Choice,
- Lookup,
- Person or Group

By default, multiple values will be separated by a comma. You may specify your own separator by setting the **multiValuesSeparator** attribute value like shown in the following query:

```

<root xmlns="http://enesyssoftware.com/schemas">
  <description>...
  <list title="Students" relativeSiteUrl="/sites/demo/"
    multiValuesSeparator="/" tableName="Students">
    <fields>Title,Skills,Spoken_x0020_language</fields>
    <query>
    </query>
  </list>
  <resultSet>Students</resultSet>
</root>

```

Running this query, you will get each student skill and spoken language separated by two slashes as shown in the following screen shot captured from the report designer:

Title	Skills	Spoken_x0020_language
Buchanan	Asp.Net//C#	English//French
Callahan	Moss 2007//Wss V3	English//German
Davolio	Asp.Net//C#//SQL Server 2005	English//Spanish
Dodsworth	Asp.Net//C#//Moss 2007//SQL Serv...	gli
Fuller	Asp.Net//C#	English//Italian
King	Asp.Net//C#	English//French//German
Leverling	Asp.Net//C#	English//French//German//Spanish//It...
Peacock	Asp.Net//C#	English//Spanish//Italian
Suyama	Wss V3	gli

Expanding multiple Values

Retrieving multiple values separated by some separator will not help if you would like to group items based on the possible values of such a column.

To address this scenario, it is possible to duplicate items for each value stored in the desired multiple values column. The desired multiple values column is specified using the `expandMultiValuesColumn` attribute as shown in the following query:

```
<list title="Students" relativeSiteUrl="/sites/demo/"
  expandMultiValuesColumn="Skills" tableName="Students">
  <fields>Title,Skills,Spoken_x0020_language</fields>
  <query>
  </query>
</list>
```

By running this query, the “Skills” column which is of type choice and holds multiple values will only get one value for each student. On the other hand, students are duplicated accordingly to reflect each skill they have as shown in the following screenshot:

Title	Skills	Spoken_x0020_...
Buchanan	Asp.Net	English,French
Buchanan	C#	English,French
Callahan	Moss 2007	English,German
Callahan	Wss V3	English,German
Davolio	Asp.Net	English,Spanish
Davolio	C#	English,Spanish
Davolio	SQL Server 2005	English,Spanish
Dodsworth	Asp.Net	gli
Dodsworth	C#	gli
Dodsworth	Moss 2007	gli
Dodsworth	SQL Server 2005	gli
Dodsworth	Wss V3	gli

Without using `expandMultiValuesColumn` attribute, you would retrieve items in the following form:

Title	Skills	Spoken_x0020_...
Buchanan	Asp.Net,C#	English,French
Callahan	Moss 2007,Wss V3	English,German
Davolio	Asp.Net,C#,SQL Server 2005	English,Spanish
Dodsworth	Asp.Net,C#,Moss 2007,SQL Server 2005,Wss V3	gli

Rather than specifying the column to expand as a literal, you may use a report parameter (e.g.: `expandMultiValuesColumn="@ChosenColumn!"`).

Important notice:

Be aware that items, being duplicated, some operations and calculations may not be appropriate when building reports using this approach.

Applying operations to lists

You are not limited to build reports based on one SharePoint list. **Enesys RS Data Extension** lets you apply specific operations between two SharePoint lists.

In fact, you can define as many `list` elements as necessary and apply as many operations as you like in order to obtain the desired Dataset from which you will build your report.

Each `list` element defined within a query returns a set of data items from a SharePoint list for which we will use the generic term "**result set**". The `tableName` attribute is used to give a unique name to the result set. The unique name of the result set will serve as the basis for specifying result sets involved in operations.

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Products" relativeSiteUrl="/sites/demo/" tableName="Products">
    <fields></fields>
    <query></query>
  </list>
</root>
```

You can apply an operation between two "result sets" by using an `sqlOp op="operation"` element. The data items resulting from an operation is considered a "result set" as well which name is given using the `dstTableName` element. Thus you can apply operations on data items resulting from other operations.

```

<sqlOp op="join">
  <dstTableName>ProdAndCat</dstTableName>
  <parentTableName>Products</parentTableName>
  <childTableName>Product categories</childTableName>
  <parentFieldName>Category</parentFieldName>
  <childFieldName>Title</childFieldName>
</sqlOp>

<sqlOp op="join">
  <dstTableName>OrdersAndDetails</dstTableName>
  <parentTableName>Orders</parentTableName>
  <childTableName>Order Details</childTableName>
  <parentFieldName>OrderID</parentFieldName>
  <childFieldName>Order_x0020_ID</childFieldName>
</sqlOp>

<sqlOp op="join">
  <dstTableName>ProdAndOrdersAndDetails</dstTableName>
  <parentTableName>OrdersAndDetails</parentTableName>
  <childTableName>ProdAndCat</childTableName>
  <parentFieldName>Product</parentFieldName>
  <childFieldName>Title</childFieldName>
</sqlOp>
<resultSet>ProdAndOrdersAndDetails</resultSet>
</root>

```

Joining result sets

The **join** operation works like the SQL inner join statement. It lets you join matching items between two result sets specified by **parentTableName** and **childTableName** based on their joining columns specified respectively by **parentFieldName** and **childFieldName** elements. The **dstTableName** element lets you give a unique name to the data items resulting from the join operation. It is possible to use this data for further operations using this unique name.

The following image displays a **join** operation between the **Products** and **Product categories** SharePoint lists (bearing that a `<list>` element has been defined for each of them). Products that do not have a matching category won't belong to the result set.

```

<sqlOp op="join">
  <dstTableName>ProdAndCat</dstTableName>
  <parentTableName>Products</parentTableName>
  <childTableName>Product categories</childTableName>
  <parentFieldName>Category</parentFieldName>
  <childFieldName>Title</childFieldName>
</sqlOp>

```

The **outerjoin** operation works like the SQL left outer join statement. It lets you join all the items from the result set specified by **parentTableName** with matching items from the result set specified by **childTableName** based on their joining columns specified respectively by

`parentFieldName` and `childFieldName` elements. There's no difference between the **join** and **outerjoin** syntax except for the `op` attribute.

```
<sqlOp op="outerjoin">
  <dstTableName>ProdAndCat</dstTableName>
  <parentTableName>Products</parentTableName>
  <childTableName>Product categories</childTableName>
  <parentFieldName>Category</parentFieldName>
  <childFieldName>Title</childFieldName>
</sqlOp>
```

You are not limited to one joining column. `parentFieldName` and `childFieldName` may contain several joining columns separated by a semi colon.

Joining lists is useful (though not limited to that usage) when dealing with SharePoint lists linked with a lookup fields. For example you might want to display products grouped by category and display the category description as well.

Merging result sets

You can merge two result set by using a **union** operation. It lets you combine the items of two result sets specified by `parentTableName` and `childTableName` elements. Both result sets must have matching fields and data types. However, two SharePoint lists can be merge even if they are not exactly identical as long as you only use shared fields in your `list` query element that are used as part of the **union** operation.

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Contacts 01" relativeSiteUrl="/sites/demo/"
    tableName="Contacts01">
    <fields>Title, Company, JobTitle, WorkCountry</fields>
    <query></query>
  </list>
  <list title="Contacts 02" relativeSiteUrl="/sites/demo/"
    tableName="Contacts02">
    <fields>Title, Company, JobTitle, WorkCountry</fields>
    <query></query>
  </list>
  <sqlOp op="union">
    <dstTableName>MergedContacts</dstTableName>
    <parentTableName>Contacts01</parentTableName>
    <childTableName>Contacts02</childTableName>
    <labelColumn>true</labelColumn>
    <parentLabelValue>Contacts01</parentLabelValue>
    <childLabelValue>Contacts02</childLabelValue>
  </sqlOp>
  <resultSet>MergedContacts</resultSet>
</root>
```

If you need to distinguish the result set's origin of each item (this would be the case if you want to build a report that group items bases on their origin), you can set the `labelColumn` child

element to true and set the `parentLabelValue` and `childLabelValue` with some specific value for each result set. In that case a specific column named `rstLabel` will be created in the final result set. The `rstLabel` column will be filled with `parentLabelValue` and `childLabelValue` element values for respectively the `parentTableName` and `childTableName` items.

Title	Company	JobTitle	WorkCountry	rstLabel
Art Braunschwe...	Split Rail Beer &...	Sales Manager	USA	Contacts01
Anabela Domin...	Tradição Hiper...	Sales Represen...	Brazil	Contacts01
Maria Anders	Alfreds Futterki...	Sales Represen...	Germany	Contacts02
Thomas Hardy	Around the Horn	Sales Represen...	UK	Contacts02
Hanna Moos	Blauer See Delik...	Sales Represen...	Germany	Contacts02
Frédérique Cite...	Blondesdsl pèr...	Marketing Mana...	France	Contacts02

Getting distinct values

The **distinct** operation lets you select unique items from the result set specified by `tableName` based on the column name specified by the `fieldName` element. The `fieldName` element might contain several column names separated by a semi colon. The optional `sortOrder` element may be used for specifying the order used to sort distinct values:

```
<sqlOp op="distinct">
  <sortOrder>DESC</sortOrder>
  <dstTableName>Distinct categories</dstTableName>
  <tableName>Products</tableName>
  <fieldName>Category</fieldName>
</sqlOp>
```

The **distinct** operation is especially useful for creating a report dataset used to set the available values of a parameter. Say you have a SharePoint Products list with a Category column and you would like to let the user of your report select at run time the category of products that will be displayed in the report. You can easily achieve this by creating a specific report dataset as shown in the following image:

```

<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This query will retrieve distinct categories from SharePoint
    "Products" list.
    Note the use of the sortOrder element that will return the
    categories in descending order.
  </description>
  <list title="Products" relativeSiteUrl="/sites/demo/"
    tableName="Products">
    <fields>Category</fields>
    <query></query>
  </list>
  <sqlOp op="distinct">
    <sortOrder>DESC</sortOrder>
    <dstTableName>Distinct categories</dstTableName>
    <tableName>Products</tableName>
    <fieldName>Category</fieldName>
  </sqlOp>
  <resultSet>Distinct categories</resultSet>
</root>

```

Getting a subset of a result set

The select operation lets you select a subset of the result set specified by `sourceTableName` based on the filtering expression specified by `selectExpression`.

```

<sqlOp op="select">
  <dstTableName>testSelect</dstTableName>
  <sourceTableName>Products</sourceTableName>
  <selectExpression>UnitPrice > 100</selectExpression>
</sqlOp>

```

The filtering expression is equivalent to the Datatable select method. Refer to the framework documentation for more details about expressions.

Though it is possible to filter SharePoint lists using CAML query, selecting a subset of a result set may be interesting in some complex queries involving several operations.

Report parameters may be used with the select expression as shown in the following example:

```

<?xml version="1.0" encoding="utf-8" ?>
<root>
  <list title="Orders" relativeSiteUrl="/sites/demo/" tableName="Orders">
    <fields></fields>
    <query>
    </query>
  </list>

  <sqlOp op="select">
    <dstTableName>testSelect</dstTableName>
    <sourceTableName>Orders</sourceTableName>
    <selectExpression>OrderDate > '@orderDate!' </selectExpression>
  </sqlOp>
  <resultSet>testSelect</resultSet>
</root>

```

Merging multiple lists

Enesys RS Data Extension provides the **multiList** query element for merging a variable number of lists using one operation.

Two approaches are available:

- The first one is based on a SharePoint list that enumerates all the lists you want to merge.
- The second approach lets you merge all lists sharing a common title within a site collection. You specify the desired approach by setting the **type** attribute to "MergingList" or "RollUpList".

Merging lists using a reference list

Use the **multiList** element with the **type** attribute set to "MergingList" for merging a set of SharePoint lists that are enumerated in a specific SharePoint list. The term "**merging list**" will be used to refer to this list.

You specify the **merging list** that is to be used by setting its **title** and **relativeSiteUrl** attributes.

The **merging list** is a SharePoint list with four mandatory columns. Each item of this list will provide information about a SharePoint list that is to be merged.

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>...
  <multiList title="Merging list" relativeSiteUrl="/sites/demo/"
            type="MergingList" tableName="Contacts">
    <fields>Title, Company, WorkCountry</fields>
    <query>
```

union	Title	relativeSiteUrl	information
Yes	Contacts 01	/sites/demo/	DemoContacts01
Yes	Contacts 02	/sites/demo/	DemoContacts02
Yes	Contacts 01 !NEW	/sites/demo/demosubsite01/	DemoContacts01FromSubsite01

The SharePoint lists that need to be merged are not necessarily completely identical as you can specify the columns that will be merged using the `fields` element. Moreover, you can specify a filter that will be applied to each list before being merged:

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>...
  <multiList title="Merging list" relativeSiteUrl="/sites/demo/"
            type="MergingList" tableName="Contacts">
    <fields>Title, Company, WorkCountry</fields>
    <query>
      <Where>
        <Eq>
          <FieldRef Name="WorkCountry" />
          <Value Type="Text">UK</Value>
        </Eq>
      </Where>
    </query>
  </multiList>
  <resultSet>Contacts</resultSet>
</root>
```

A specific column named **"rstLabel"** is added to the result set obtained from the `multiList` query element. This column will be filled with the value of the **merging list's** column **Information** depending on the list from which the item is retrieved:

Dataset: MergedContacts

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This query will merge all SharePoint lists enumerated
    in the list "Merging list" located on "/sites/demo/".
  </description>
  <multiList title="Merging list" relativeSiteUrl="/sites/demo/"
    type="MergingList" tableName="Contacts">
    <fields>Title, Company, WorkCountry </fields>
    <query>
      <Where>
        <Eq>
          <FieldRef Name="WorkCountry" />
          <Value Type="Text">UK </Value>
        </Eq>
      </Where>
    </query>
  </multiList>
  <resultSet>Contacts </resultSet>
</root>
```

Title	Company	WorkCountry	rstLabel
Ann Devon	Eastern Co...	UK	DemoContacts01
Elizabeth Brown	Consolidate...	UK	DemoContacts02
Helen Bennett	Island Trading	UK	DemoContacts01FromSubsite01
Hari Kumar	Somen Seas...	UK	DemoContacts01FromSubsite01

Merging list

This list contains a set of SharePoint lists that can be merged using the <multiList> query element provided by Enesys RS Data Extension.

New Actions Settings View: All Items

union	Title	relativeSiteUrl	information
Yes	Contacts 01	/sites/demo/	DemoContacts01
Yes	Contacts 02	/sites/demo/	DemoContacts02
Yes	Contacts 01 !NEW	/sites/demo/demosubsite01/	DemoContacts01FromSubsite01

Rolling up lists in a site collection

Use the `multiList` element with the `type` attribute set to `RollUpList` for merging lists having the same "title" within an entire site collection.

The following query will merge all SharePoint lists named "Contacts 01" within the entire `/sites/demo/` site collection:

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This query will merge all SharePoint lists which title is
    "Contacts 01" within "/sites/demo/" site collection.
  </description>
  <multiList title="Contacts 01" relativeSiteUrl="/sites/demo/"
    type="RollUpList" tableName="Contacts">
    <fields>Title, Company, WorkCountry</fields>
    <query></query>
  </multiList>
  <resultSet>Contacts</resultSet>
</root>
```

You are not obliged to merge lists starting from the root of the site collection. It is possible to specify a child site as the starting point of the merging process (e.g.: `relativeSiteUrl="/sites/demo/demosubsite01/"`).

When running such a query, a column named "rstLabel" will be added to the result set. This column will contain the relative site Url of the list from which the items have been retrieved as shown in the following screenshot where the previous query is run from the report designer:

Dataset: MergedContacts Command type: Text

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This query will merge all SharePoint lists which title is
    "Contacts 01" within "/sites/demo/" site collection.
  </description>
  <multiList title="Contacts 01" relativeSiteUrl="/sites/demo/"
    type="RollUpList" tableName="Contacts">
    <fields>Title, Company, WorkCountry</fields>
    <query></query>
  </multiList>
  <resultSet>Contacts</resultSet>
</root>
```

Title	Company	WorkC...	rstLabel
Helvetius Nagy	Trail's Head Go...	USA	/sites/demo/demosubsite01/
Karl Jablonski	White Clover M...	USA	/sites/demo/demosubsite01/
Thomas Hardy	Around the Horn	UK	/sites/demo/demosubsite01/demosubsubsite01/
Victoria Ashworth	B's Beverages	UK	/sites/demo/demosubsite01/demosubsubsite01/
Yang Wang	Chop-suey Chin...	Switzer...	/sites/demo/demosubsite01/demosubsubsite01/
Sven Ottlieb	Drachenblut Del...	Germany	/sites/demo/demosubsite01/demosubsubsite01/
Yoshi Latimer	Hungry Coyote ...	USA	/sites/demo/demosubsite01/demosubsubsite01/
Yoshi Tannamuri	Laughing Bacch...	Canada	/sites/demo/demosubsite01/demosubsubsite01/
Simon Crowther	North/South	UK	/sites/demo/demosubsite01/demosubsubsite01/
Yvonne Moncada	Océano Atlántic...	Argentina	/sites/demo/demosubsite01/demosubsubsite01/
Sergio Gutiérrez	Rancho grande	Argentina	/sites/demo/demosubsite01/demosubsubsite01/
Zbyszek Piestrz...	Wolski Zajazd	Poland	/sites/demo/demosubsite01/demosubsubsite01/
Maria Anders	Alfreds Futterki...	Germany	/sites/demo/demosubsite02/

Rather than specifying the starting site in the query, you may use a report parameter (e.g.: `relativeSiteUrl="@StartingSiteUrl!"`).

Retrieve lists information, groups and permissions

Enesys RS Data Extension provides specific query elements for retrieving sites and lists information such as list collection, list permissions, web permissions, etc.

List collection

The `listCollection` query element lets you retrieve information about all the lists in a site or a set of sites.

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <listCollection relativeSiteUrl="/sites/demo/" includePermissions="true"
    tableName="lists">
  </listCollection>
</root>
```

Optionally, you can retrieve the permissions as well for each list by setting the `includePermissions` attribute to `true`.

List permissions

The `listPermissions` query element lets you retrieve permissions from a specific list.

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <listPermissions listID="@List!" relativeSiteUrl="@SiteUrl!" tableName="list">
  </listPermissions>
</root>
```

Parameters may be used for the `listID` and `relativeSiteUrl` attributes values.

Web permissions

The `webPermissions` query element lets you retrieve the permissions given at the site level for a specific site or a set of sites.

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <description>
    This query will return permissions for the specific site passed
    as a report parameter.
    Details about users within SharePoint groups will be provided
    (expandGroups="true").
  </description>
  <webPermissions relativeSiteUrl="@SiteUrl!" expandGroups="true"
    tableName="perms">
  </webPermissions>
  <resultSet>perms</resultSet>
</root>
```

Optionally, you can retrieve the users (or AD Groups) belonging to each group by setting the `expandGroups` attribute to `true`.

SharePoint groups

The `webGroups` query element lets you retrieve SharePoint groups from a site or a set of sites.

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <webGroups relativeSiteUrl="@SiteUrl!" expandGroups="true"
    tableName="groups">
  </webGroups>
  <resultSet>groups</resultSet>
</root>
```

Optionally, you can retrieve the users (or AD Groups) within each group by setting the `expandGroups` attribute to `true`.

Specifying a set of sites

Except `listPermissions`, all those commands are sharing a common approach for specifying sites for which they apply.

You can specify the sites to handle in one of three ways:

By writing directly the relative site Url of the site (e.g.: `/sites/demo/`) within the query:

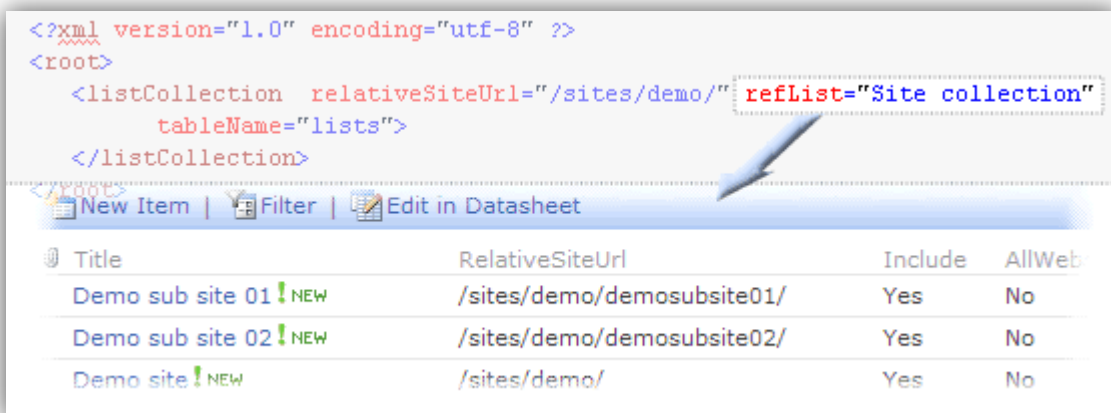
```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <listCollection relativeSiteUrl="/sites/demo/" tableName="lists">
  </listCollection>
</root>
```

By using a report parameter so that the user might select the site when running the report:

```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <listCollection relativeSiteUrl="@SiteUrl!" tableName="lists">
  </listCollection>
</root>
```

By specifying a SharePoint list that contains the list of sites to handle:

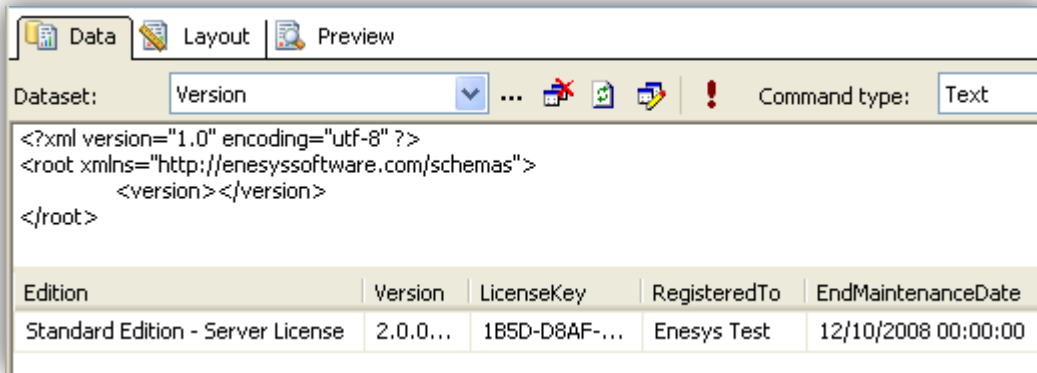
```
<?xml version="1.0" encoding="utf-8" ?>
<root>
  <listCollection relativeSiteUrl="/sites/demo/" refList="Site collection"
    tableName="lists">
  </listCollection>
</root>
```



Title	RelativeSiteUrl	Include	AllWeb
Demo sub site 01 !NEW	/sites/demo/demosubsite01/	Yes	No
Demo sub site 02 !NEW	/sites/demo/demosubsite02/	Yes	No
Demo site !NEW	/sites/demo/	Yes	No

Retrieving Enesys RS Data Extension version information

You can easily get version and license information by using the `version` query element as shown in the following screenshot:



Sample reports

The "ReportSamples" folder provided as part of the installation package includes report samples as well as a backed up SharePoint site containing sample lists used to run the sample reports.

Files details

File/folder	Description
sites-demo.bak	<p>Backed up SharePoint site collection containing sample sites and lists used to run the sample reports. The site can be restored using the following STSADM command:</p> <pre>STSADM.EXE -o restore -url http://localhost/sites/demo -filename \ReportSamples\sites-demo.bak</pre> <p>If possible, keep demo as the restored site name so that you will not have to modify the "relativeSiteUrl" attribute for each sample report query.</p> <p>Please note that if you are running a non English SharePoint Server, you will need to install the English Language template pack before installing the demo site.</p>
ReportSamples2005	This folder contains the sample reports in the form of a report designer solution for SQL Server 2005.

Sample reports

The following reports are provided as part of the solution:

Report	Description
Dependant parameters	This report shows how define a parameter that depends on the value of another one.
IssuesGraphicalStats	Report file for the sample "Building a report showing graphical stats from a SharePoint Issues List". Complete Instructions for

	building this report is available on our web site.
IssuesGraphicalStats By Component	Report file for the sample "Using report parameters with SharePoint lists". Complete Instructions for building this report is available on our web site.
SalesByCategory	Report file for the sample "Joining SharePoint lists". Complete Instructions for building this report is available on our web site.
YearlySalesByCategory	Report file for the sample "Joining SharePoint lists". Complete Instructions for building this report is available on our web site.
IT – List Collection from sites in a reference list	Display the List collection and their permission for a set of sites enumerated by the SharePoint list "Site collection" located on /sites/demo/.
IT - List permissions	Display permissions for a specific list. This report is not intended (though it's possible) to be run directly but rather from other reports.
IT - Lists and permissions for a specific site	Display lists and lists' permissions for a specific site selected when running the reports.
IT – SharePoint Groups and Users for a site	Display SharePoint groups as well as users belonging to each group for a selected site collection. When running the report, the site is selected amongst sites specified in the SharePoint list "Site collection" located on the site "/sites/demo/"
IT – Web permissions for a specific site and subsites	Display permissions (at the site level) for a specific site selected when the report is run. SharePoint Groups are expanded for getting user belonging to each group.
IT – Web permissions	Display web permissions for a specific site. This report is not really intended to be run directly but rather from other reports.
Recurring Events Real World	This report shows how to retrieve events between 2 dates – both recurring and non recurring.
Stripping Html	Sample report showing the effect of stripping html using <code>stripHtml</code> attribute.
Running Values Sample	Sample report demonstrating how to add a custom column holding a running sum.
Select Operation with Parameter	This report demonstrates how to select a subset of a result set using a report parameter.
Version Information	This report let's you retrieve version information and License key of Enesys RS Data Extension.

The following reports provided as part of the report designer solution don't have any report layout. They are intended to provide specific query syntax examples:

Report	Description
Query - Distinct values	Query to retrieve distinct values from a SharePoint list.
Query - Expanding Multi values Column using parameter	Query demonstrating how to expand a multiple values column by passing the name of the column as a report parameter.
Query - Expanding Multi values	Query demonstrating how to expand a multiple values column.
Query - Lists and folders	Query demonstrating how to use the folder attributes for retrieving root folders from as specific list.
Query - Merging Lists	Query demonstrating the use of the "union" operator to merge two lists.
Query - Merging Multiple Contact Lists	Query demonstrating the use of the multiList query element for merging multiple lists. The lists that must be merged are enumerated in a specific SharePoint list.
Query - Multi values specific separator	Query showing how to use a specific separator for cleaning multiple values columns.
Query - Pending Items	Query for retrieving pending items from a list.
Query - Relative Site Url as a parameter	Query showing how to retrieve data from a SharePoint list which site Url is specified using a report parameter.
Query - Rolling up Contact Lists specifying starting site using a parameter	This query demonstrates how to roll up lists within an entire site collection. The starting site from which lists will be rolled up is specified using a report parameter.
Query - Rolling up Contact Lists	This query demonstrates how to roll up lists within an entire site collection.
Query - Row Limit	Sample query showing the use of the rowLimit attribute.
Query - Simple Report	Query for retrieving products that belongs to a specific category. The category is specified at run time by the user running the report.
Query - Selecting a subset	Sample query demonstrating the use of "select" operation to get a subset of a result set.
Query - Specific folder	Query for retrieving items and sub-folders in a specific folder.

Data source credentials

In order to retrieve SharePoint lists data, credentials information must be passed to SharePoint Web Services.

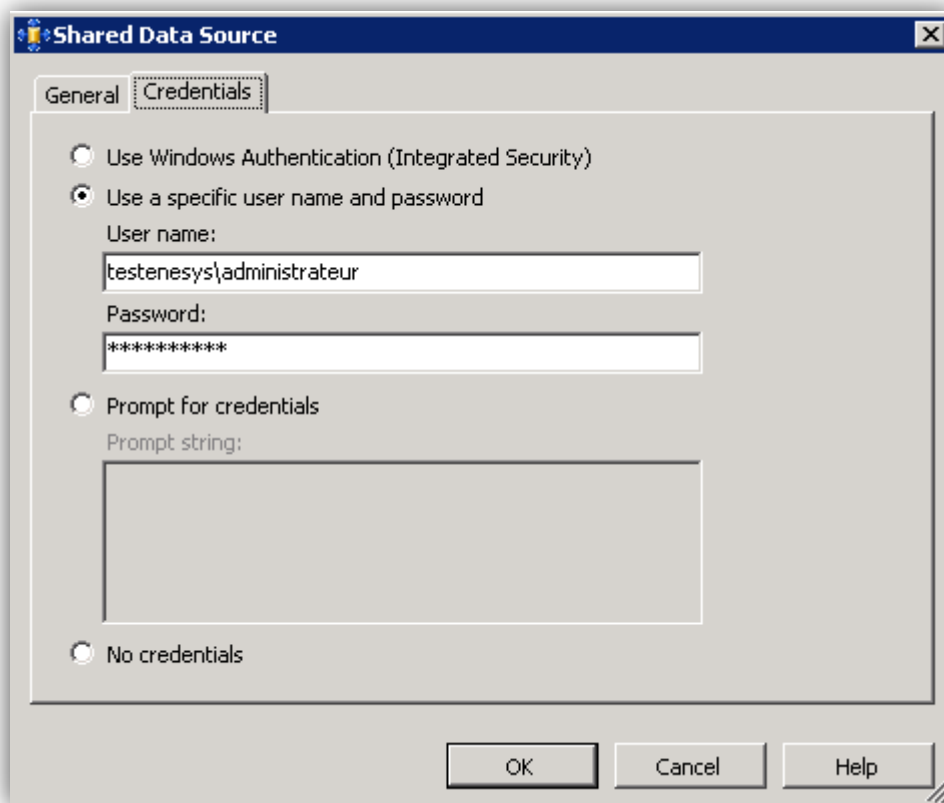
This chapter will explain the various credential options provided by Reporting Services and their effect on Enesys RS Data Extension.

For clarity, we will differentiate Report Designer and Report Server side.

Report Designer

When designing a report, you will use one or more datasets connected to a data source. Though it is possible to embed data source information within a report, we recommend using shared data sources for making future modifications easier.

For connecting to the data source, you must provide credentials information using the credential tab as shown in the following image:



Use Windows Authentication (Integrated Security)

When you use **Windows Integrated Security**, the credentials of the user currently designing the report will be passed to SharePoint Web Services. You will need the necessary rights on the SharePoint lists that will be retrieved using this data source.

Use a specific name and password

When using this option, Enesys RS Data Extension will create network credentials for the account specified and will pass those credentials to SharePoint Web Services. The account specified must have the necessary rights for accessing the SharePoint lists used in the report.

Prompt for credentials

This option will only work in preview mode and will let you specify a run time the account that should be used to connect to SharePoint.

We do not recommend this option when designing a report as this will not work when you run the query in the data view.

No credentials

No credentials is not an option for Enesys RS Data Extension.

Report Server

When deploying a report to the server, several cases must be considered.

If you have embedded the data source information within the report, you will end up with the same data source configuration on the server. At this stage, you may decide to use a shared data source or change the report-specific data source connection string or credentials.

If your report is using a shared data source in the report designer, the shared data source will be deployed on the server along the report unless a shared data source with the same name already exists in the server deployment path (unless you have configured your project to overwrite data sources). This is something important to note as you may end up with a completely different data source configuration once a report is deployed on the server. At this stage, you may configure the report to use a different data source or even create a report-specific data source configuration though we would not recommend this approach unless you have specific reasons to make it so.

Credentials supplied by the user running the report

When using this option, the user running the report will be prompted to enter a user name and password. The credentials of the user account entered will be passed to SharePoint Web Services.

Connect using:

Credentials supplied by the user running the report

Display the following text to prompt user for a user name and password:

Type or enter a user name and password to access the data sou

Use as Windows credentials when connecting to the data source

Credentials stored securely in the report server

User name:

Password:

Use as Windows credentials when connecting to the data

It is not necessary to check "Use as Windows credentials when connecting to the data source" as Enesys RS Data Extension will use network credentials anyway. This option will only makes a difference with data sources that may use different authentication schemes like SQL-Server.

Credentials stored securely in the report server

When using this option, the credentials of the user account entered and stored in the server will be passed to SharePoint Web Services.

Connect using:

Credentials supplied by the user running the report

Display the following text to prompt user for a user name and password:

Type or enter a user name and password to access the data source

Use as Windows credentials when connecting to the data source

Credentials stored securely in the report server

User name: testenesys\john.smith

Password: ●●●●●●●●●●●●●●●●

Use as Windows credentials when connecting to the data source

Impersonate the authenticated user after a connection has been made to the data source

Windows integrated security

Credentials are not required

It is not necessary to check “Use as Windows credentials when connecting to the data source” as Enesys RS Data Extension will use network credentials anyway. This option will only makes a difference with data sources that may use different authentication schemes like SQL-Server. Note however that checking this option will work properly though it will make a difference on how credentials are passed to Enesys RS Data Extension by the report server.

The “Impersonate the authenticated user after a connection has been made to the data source” is meaningless for Enesys RS Data Extension and will not be used whether you checked it or not.

Windows integrated security

When you use **Windows Integrated Security**, the credentials of the user running the report will be passed to SharePoint Web Services.

Impersonate the authenticated user after a connection has been made to the data source

Windows integrated security

Credentials are not required

Be aware that if Reporting Services is not on the same machine as SharePoint, you may need to deploy Kerberos delegation in order to pass credentials from the Report Server to SharePoint.

Credentials are not required

When using this option, the credentials of the unattended execution account will be used if it is configured.

This not a recommended option when using Enesys RS Data Execution.

Which credentials should you use?

Obviously, there is no definitive answer and it may depend on how you are organized.

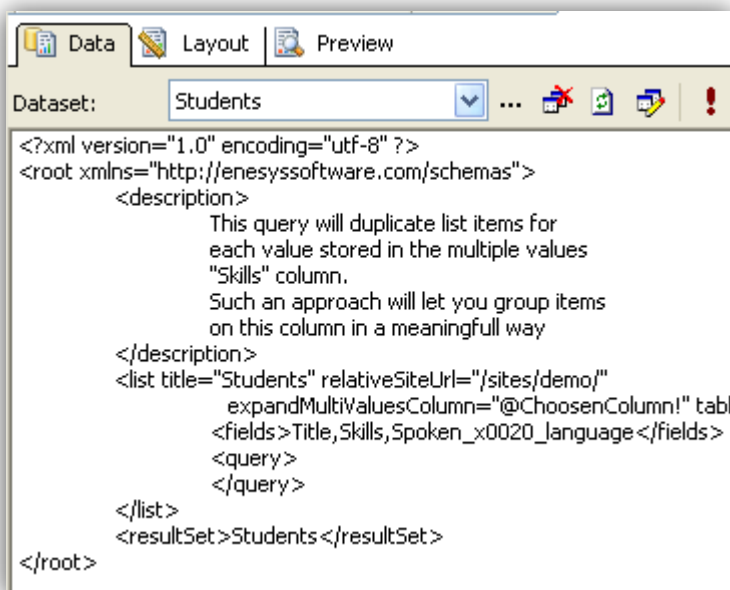
However, from our experience on customer site, our preference goes to “credentials stored in the report server” for the following reasons:

- Stored credentials are a requirement for reports that run on a schedule (subscriptions).
- We will configure access security at the report or report server folder level so that users are not polluted by reports they won't be able to run.
- It is sometimes desirable to allow users run a report on a SharePoint list for which they don't have permissions. SharePoint doesn't offer the possibility to give read permissions on a subset of list's columns. Suppose you have a SharePoint contact list that you would like being available by everybody except for the home phone number that should only be available to managers. Unless you are ready to maintain two SharePoint lists, you may have one list for which only managers have permissions. For other users, you could build a report that would not display the home phone number column and deploy this report using stored credentials and give all users access rights to run the report.

Of course, you may use a mix of the credentials options available in order to achieve the same goals.

Using Intellisense for writing queries

The generic query designer used to write dataset queries within the report designer Data view does not support any form of syntax coloring or helper tools for building queries. It's in fact a rather basic text editor.

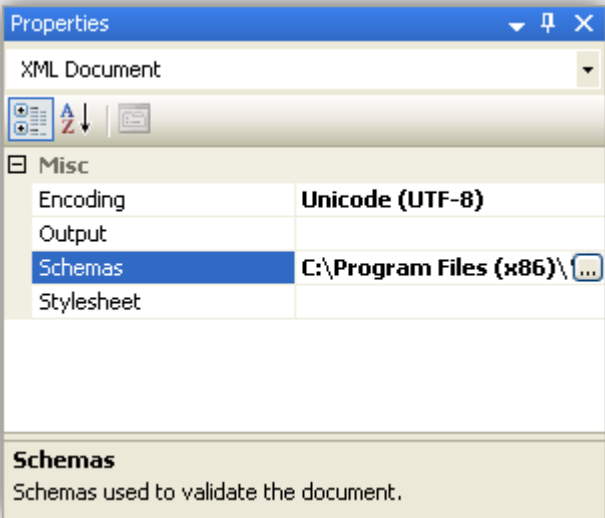


To make things somewhat easier for writing Enesys RS Data Extension queries, you can add an xml file to your report project and use this xml file for writing queries using the xml aware editor of Visual Studio.

You can make things even more comfortable by referencing ERSDE schema and benefit from intellisense features.

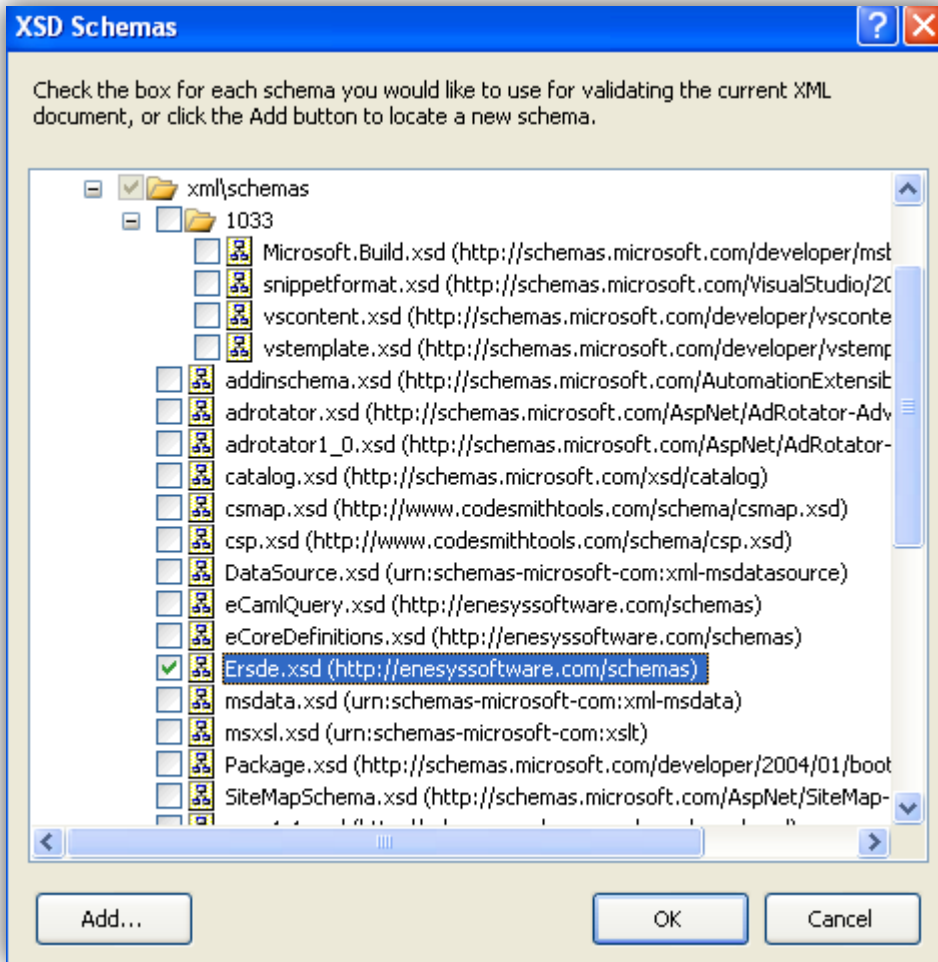
Though not an absolute necessity, we recommend installing ERSDE schema files along other BI development studio (VS2005) schema files. See “Installing schema files” page “12” for more information.

Once you have added an xml file to your project, specify the schema used to validate your xml document in the properties section:

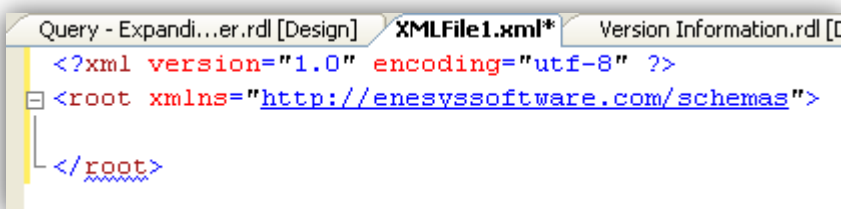


Click on the button next to the **Schemas** property.

In the opening dialog (see screenshot below), select Ersde.xsd schema and click ok.



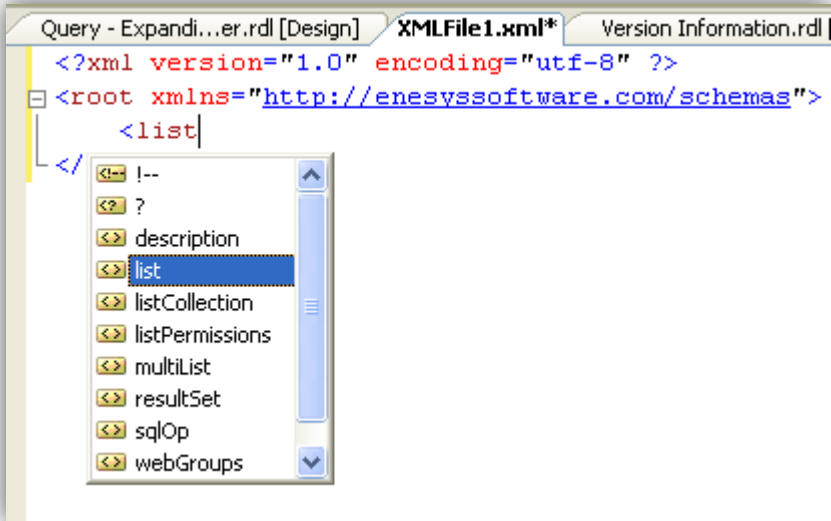
Now in your xml file, write a root element. Ersde Schema namespace will be automatically added as shown in the following screenshot:



You will automatically be provided with intellisense support while going on writing the query.

When you are done with writing the query, you can just copy it to the generic query designer where it can be run and tested.

Note that queries written for versions of Enesys RS Data Extension prior to 2.0 were not supporting any schema. If you are copying such a query from a report into the xml file for editing purpose, you will need to add ERSDE namespace to the root element of the query for getting the benefit of intellisense and xml validation.



Reference

list element

Element/Attribute	Description
title	Title of the SharePoint list from which you want to retrieve data.
listID	<p>GUID of the SharePoint list from which you want to retrieve data. Though not specifically self describing, the list GUID has the advantage of not changing over time even if the list title is modified.</p> <p>Note that if you specify both title and listID attributes, the later will take precedence.</p>
relativeSiteUrl	<p>URL of the SharePoint site containing the list. This URL is relative to the SharePoint site as it is defined in the data source. This approach was chosen to make it possible to easily move from a SharePoint test server to a SharePoint production server by simply modifying the data source.</p> <p>The relativeSiteUrl attribute may be specified using a report parameter (e.g.: relativeSiteUrl=" @url!").</p>
tableName	<p>This attribute makes it possible to assign a name to the set of data retrieved from your SharePoint list.</p> <p>Though not extremely useful (but still mandatory) when you retrieve data from a single list, this name will be the basis for applying sql-like operations between SharePoint lists.</p>
moderationType	<p>This optional attribute makes it possible to retrieve pending items for lists for which content approval is required.</p> <p>By setting the value of the attribute to "Moderator", you will retrieve pending and rejected items as well as approved items.</p>
useDisplayName	<p>Optional Boolean. This attribute lets you indicate that you want to use the display names of the columns rather than their internal names as part of the CAML format query and the fields element.</p> <p>SharePoint lists columns have both an internal name and a display name. When a column is initially created, the display name and the internal name are identical (except if there are spaces or accent marks). However, the internal name is never modified—even if you change the name of the column. Thus you may end up with a column whose internal name no longer has any connection to the name displayed.</p> <p>CAML queries use the columns' internal names. To avoid having to</p>

Element/Attribute	Description
	<p>search for the internal name associated with the columns using a complementary tool (CAML Builder or other), you can use the column display names by setting the attribute to "true". Before performing the SharePoint Web Service query, the display names will be automatically replaced by the internal names. If you prefer using the internal names of the columns you can set the attribute to "false" or even not define the attribute.</p>
expandRecurrent	<p>Optional Boolean. If set to "true", recurring events will be expanded based on their recurrence definition. This attribute will not have any effect when used with lists other than event's ones.</p> <p>The default value is "false".</p>
expandFirstDate expandLastDate	<p>Optional. Range of dates for which the recurring events will be expanded. The attributes have no effect if expandRecurrent is false or not defined.</p> <p>The date format must respect the following format :</p> <p>AAAA-MM-JJTHH:MM:SS.</p> <p>A report parameter may be used instead of a literal. The report parameter used must be either of type Datetime (ERSDE will automatically convert it to the appropriate format) or a string respecting the date format as shown above.</p>
stripHtml	<p>Optional Boolean. When set to "true", specifies html tag should be stripped from SharePoint columns containing html.</p> <p>The default value is "false".</p>
multiValuesSeparator	<p>Optional. Specify a string that will be used as a separator for multiple values columns: Choice, lookup and Person or Group.</p> <p>The default separator is a comma.</p>
expandMultiValuesColumn	<p>Optional. Specify the name of a multiple values column.</p> <p>Each list item will be duplicated for each value stored in the column ending with a column holding a single value.</p> <p>This approach lets you use Reporting Services grouping features to get items into several groups.</p> <p>Instead of specifying the name of the column as a literal, you may use a report parameter.</p> <p>See "Expanding multiple Values" page 36 for more information about using this feature.</p>
folder	<p>Optional. Specify a specific folder from which to return items.</p> <p>The default behavior of the list element is to return all items of the</p>

Element/Attribute	Description
	list without any consideration for folders.
	When specifying a folder, you will retrieve items from that folder only. Note that you will also retrieve subfolders of this specific folder. You can discern between items and folders using the FSObjType column which will be set to "1" for folders and "0" for items.
	You can retrieve root items and folders by setting folder value to "/" (slash).
	A report parameter may be used to specify the folder attribute.
rowLimit	Optional. Specify the number of items returned by the list query element.
	Using rowLimit attribute may be useful when you want to build a report showing most important items (e.g.: top sales, ...). Of course, You will have to order the list accordingly using the appropriate CAML query.

fields element

The **fields** element lets you specify the columns defined in the context of this element.

Each column name must be separated by a comma. Depending on the value of the attribute **useDisplayName**, the column's display name or internal name must be used.

If you do not enter any columns, all of the list's columns will be returned.

query element

SharePoint CAML query. Please see the SharePoint documentation for information about CAML query syntax.

Some additional functionalities have been added to better integrate its use with Reporting Services:

If you have set **useDisplayName** attribute to **true**, you will need to use column display names instead of internal names as it is normally the case with CAML queries.

Parameters provided by Reporting Services may be used within the query element. Each parameter takes the following form: **@name_parameter!** (a name surrounded by @ and !).

customFields element

The **customFields** element is a container for **field** elements used for specifying columns holding a running value (future versions of ERSDE may add other types of columns).

The following table describes the attributes of the **field** element.

Element/Attribute	Description
Name	Name of the custom column to be added to the result set.

Element/Attribute	Description
dataType	Data type of the column that will be holding the running value (System.Int32, System.Decimal,).
Op	Function used for calculating the running values. The following values may be used: <ul style="list-style-type: none"> • Sum • Min • Max
groupColumnName	Optional. Specify the name of the column for which the running value will be reset each time its value changes.
Param	Name of the column containing the value.

multiList element

The **multiList** element lets you merge multiple SharePoint lists using one operation.

Two different approaches are available for merging lists:

- The first one is based on a SharePoint list that enumerates all the lists you want to merge.
 - The second approach lets you merge all lists sharing a common title within a site collection.
- You specify the desired approach by setting the **type** attribute to “**MergingList**” or “**RollUpList**”.

Basically, the **multiList** element has the same attributes and child elements as the **list** element. However, some attributes have a different meaning when used in **multiList** element context. Also, the **multiList** element has a **type** attribute not available with the **list** element.

We will only describe attributes that have a different meaning than when used with the **list** element. Refer to the “list element” page “59” for further information on the other attributes available.

Merging lists approach

When setting the **type** attribute to **MergingList**, the **multiList** element will let you specify a SharePoint list that will enumerate all the lists you would like to merge.

Note that a SharePoint list template is provided for the merging list (Merging List.stp) in the ReportSamples directory. You can upload this template in the desired template library and create a new list from this template.

The following table describes the meaning of the attributes when using this approach.

Attribute	Description
Title	Title of the SharePoint list that contains the definition of the lists that are to be merged (merging list).

Attribute	Description
relativeSiteUrl	Url of the SharePoint site containing the merging list. A report parameter may be used for specifying the Url.
tableName	Name assigned to the data resulting from operation.

Rolling up lists approach

When setting the **type** attribute to **RollUpList**, the **multiList** element will let you merge all lists sharing a common **title** within a site collection.

The following table describes the meaning of the attributes when using this approach.

Attribute	Description
Title	Title of the SharePoint list that is to be merged. Any list having this title within the entire site collection will be merged as a result of the operation.
relativeSiteUrl	Url of the starting SharePoint site from which lists should be merged. Lists from the site specified by relativeSiteUrl attribute and its child sites (recursively) having a title like specified by the title attribute, will be merged together as the result of the operation. A report parameter may be used for specifying the value of the attribute.
tableName	Name assigned to the data resulting from the multiList operation. Using this name, you may apply further operation to this data.

Merging list columns

The definition of the lists to be merged must be populated in a SharePoint list. It is this list that is designated by the **title** attribute in the **multiList** element having a **type** attribute set to **MergingList** element.

Each record in the list will represent the definition of a SharePoint list to be merged.

The list must contain the following columns:

Column - type	Description
title - text	Title of the SharePoint list that is to be merged during the multiList operation
relativeSiteUrl - text	Relative URL of the SharePoint site containing the list (e.g. /sites/demo/).
information - text	To make it possible to distinguish the origins of the combined data, the content of this column is added to each record in each of the lists as part of the "rstLabel" column that is systematically added to the

Column - type	Description
	result set.
union - Yes/No	Only the lists for which this field is "Yes" will be merged. Using such a column makes it possible to avoid deleting and re-creating the definition of lists each time you want to temporarily restrict the operation to a sub-set of the lists.

Note that a SharePoint list template is provided for the merging list (Merging List.stp) in the ReportSamples directory. You can upload this template in the desired template library and create a new list from this template rather than from scratch.

sqlOp element

The `sqlOp` element lets you apply Sql-like operations between two result sets. A result set is any data returned by an Enesys RS Data Extension query element (`list`, `multiList`, `sqlOp`,...).

Op= "join"

This operation performs a join between two lists. The parameters are:

Element	Description
<code>dstTableName</code>	Name of the destination table for the join operation. The result of the operation can be used as part of another operation by invoking this name.
<code>parentTableName</code>	Parent list used to execute the join operation. It is also possible to use the result of another operation.
<code>childTableName</code>	Child table. It is also possible to use the result of another operation.
<code>parentFieldName</code>	Columns used to perform the join in the parent table. Each column must be separated by a comma.
<code>childFieldName</code>	Columns used to perform the join in the child table. Each column must be separated by a comma.

Op= "outerjoin"

This operation performs an outer join between two lists. The parameters are identical to those in the "join" operation.

Op= "union"

This operation performs a union between two lists. These are the parameters:

Element/Attribute	Description
<code>dstTableName</code>	Name of the destination table for the union operation. The result of the operation can be used as part of another operation by invoking this name.
<code>parentTableName</code>	Parent list. It is also possible to use the result of another operation.

Element/Attribute	Description
<code>childTableName</code>	Child list. It is also possible to use the result of another operation.
<code>labelColumn</code>	Indicates if you want to use a specific column that would make it possible to distinguish the records for each table. If <code>labelColumn</code> has a "true" value, a column bearing the name "rstLabel" will be automatically created in the result of the "union" operation. This column will be filled with the value of the <code>parentLabelValue</code> element for the parent table records and with the value of the <code>childLabelValue</code> element for the child table records. If <code>labelColumn</code> has a "false" value, no column will be created and the <code>parentLabelValue</code> and <code>childLabelValue</code> elements will not be used.
<code>parentLabelValue</code>	Value used to distinguish the parent table records.
<code>childLabelValue</code>	Value used to distinguish the child table records.

Op= "distinct"

This operation makes it possible to obtain the distinct elements of a list based on specific columns. The parameters are:

Element/Attribute	Description
<code>dstTableName</code>	Name of the destination table for the distinct operation. The result of the operation can be used as part of another operation by using this name.
<code>tableName</code>	List for which you want distinct values. It is also possible to use the result of another operation.
<code>fieldName</code>	Columns making it possible to determine distinct values. Each column must be separated by a comma.

Op="select"

Element	Description
<code>dstTableName</code>	Name of the destination table (or result set) that will contain a subset of the result set represented by <code>sourceTableName</code> .
<code>sourceTableName</code>	Source table (or result set).
<code>selectExpression</code>	Filter expression.

listCollection element

The `listCollection` query element lets you retrieve information about all the lists in a site, a site collection or a set of sites you specify using a reference list.

Attribute	Description
<code>relativeSiteUrl</code>	Specify the relative Url of the site from which you would like to retrieve the list collection.
<code>refList</code>	<p>Optional. The <code>refList</code> attribute is used to specify a SharePoint list that contains an enumeration of all the sites from which you would like to retrieve list collection.</p> <p>When using the <code>refList</code> attribute, the <code>relativeSiteUrl</code> attribute will not specify anymore the site from which to retrieve the list collection but rather the site where is located the SharePoint list specified by the <code>refList</code> attribute.</p> <p>Note that a SharePoint list template is provided for the reference list (Reference List.stp) in the ReportSamples directory. You can upload this template in the desired template library and create a new list from this template.</p>
<code>allWebs</code>	<p>Optional Boolean. Setting <code>allWebs</code> to true will return the list collection from all the sites of the site collection to which the site specified by <code>relativeSiteUrl</code> attribute belongs to.</p> <p>Default value is <code>false</code>.</p>
<code>includePermissions</code>	<p>Optional Boolean. When set to <code>true</code>, permissions for each list will be returned as well.</p> <p>Default value is <code>false</code>.</p>
<code>expandGroups</code>	<p>Optional Boolean. When set to <code>true</code>, it will expand SharePoint groups so that individual user or Active Directory group are returned.</p> <p>Please note that this attribute has no effect if <code>includePermissions</code> attribute is not set to <code>true</code>.</p> <p>Default value is <code>false</code>.</p>
<code>tableName</code>	<p>Name assigned the set of data retrieved from the <code>listCollection</code> query element.</p> <p>This name will be used as the basis for applying sql-like operations between result sets.</p>

listPermissions element

The `listPermissions` query element lets you retrieve permissions for a specific SharePoint list.

Attribute	Description
tableName	Name assigned the set of data retrieved from the listCollection query element. This name will be used as the basis for applying sql-like operations between result sets.
relativeSiteUrl	Specify the relative Url of the site where is located the SharePoint list.
listID	ID of the SharePoint list.
expandGroups	Optional Boolean. When set to true , it will expand SharePoint groups so that individual user or Active Directory group are returned. Default value is false .

webPermissions element

The **webPermissions** query element lets you retrieve the permissions given at the site level for a specific site, all sites within a site collection or a set of sites you specify using a reference list.

Attribute	Description
relativeSiteUrl	Specify the relative Url of the site from which you would like to retrieve assigned permissions.
refList	Optional. The refList attribute is used to specify a SharePoint list that contains an enumeration of all the sites from which you would like to retrieve permissions. When using the refList attribute, the relativeSiteUrl attribute will not specify anymore the site from which to retrieve the permissions but rather the site where is located the SharePoint list specified by the refList attribute. Note that a SharePoint list template is provided for the reference list (Reference List.stp) in the ReportSamples directory. You can upload this template in the desired template library and create a new list from this template.
allWebs	Optional Boolean. Setting allWebs to true will return permissions from all the sites of the site collection to which the site specified by relativeSiteUrl attribute belongs to. Default value is false .
expandGroups	Optional Boolean. When set to true , it will expand

Attribute	Description
	SharePoint groups so that individual user or Active Directory group are returned. Default value is false .
tableName	Name assigned the set of data retrieved from the webPermissions query element. This name will be used as the basis for applying sql-like operations between result sets.

webGroups element

The **webGroups** query element lets you retrieve SharePoint groups from a site or a set of sites.

Attribute	Description
relativeSiteUrl	Specify the relative Url of the site from which you would like to retrieve SharePoint groups. Though you may specify a sub site within a site collection, be aware that SharePoint groups are defined at the site collection level. Thus you will get the same result by specifying any site belonging to the same site collection.
refList	Optional. The refList attribute is used to specify a SharePoint list that contains an enumeration of all the sites from which you would like to retrieve SharePoint groups. When using the refList attribute, the relativeSiteUrl attribute will not specify anymore the site from which to retrieve SharePoint groups but rather the site where is located the SharePoint list specified by the refList attribute. You refList should only not contain sites from the same site collection as SharePoint groups are defined at the site collection level. Note that a SharePoint list template is provided for the reference list (Reference List.stp) in the ReportSamples directory. You can upload this template in the desired template library and create a new list from this template.
expandGroups	Optional Boolean. When set to true , it will expand SharePoint groups so that individual user or Active Directory group are returned.

Attribute	Description
tableName	<p>Default value is false.</p> <p>Name assigned the set of data retrieved from the webGroups query element.</p> <p>This name will be used as the basis for applying sql-like operations between result sets.</p>

version element

The **version** element sole purpose is to return information about your version of **Enesys RS Data Extension**.

```
<?xml version="1.0" encoding="utf-8" ?>
<root xmlns="http://enesyssoftware.com/schemas">
  <version></version>
</root>
```

The **version** element takes precedence over any other query elements defined.

The **version** element has no attributes.

By running such a query (report) from within the report designer, you will only get version information for the extension installed along your Business Intelligence Development studio. To get information regarding the extension installed on your server, you will have to deploy the report on your server and run it from there using the web interface (or any other method you would like).

resultSet element

The **resultSet** element lets you specify which result set should be returned to Reporting Services.

Your query may be composed of various intermediate result sets. Obviously you would likely want to return the result of your last operation to Reporting Services. However, for testing purposes, it may be useful to return data from an intermediate result set.

Most query elements have a **tableName** attribute for naming to the resulting data. An exception to this is the **sqlOp** element that uses the **dstTableName** child element for naming the resulting data.

Support

You may obtain support by either using the forum on our web site (<http://www.enesyssoftware.com>) or by sending an email to support@enesys.fr.

Please provide as much information as possible to describe your problem when asking for support:

- Server configuration,
- Error,
- Query used,
- ...