Filtering and Sorting Data Lists

# Introduction

Filtering, sorting data is something all web data systems need to do. The usual method is to place a static call in the code to ensure we only return the data we want and in the order we want.

This traditional method can be easily done in the Rocket Framework and in some cases could be the preferred method. However, it doesn’t lead to reuse and flexibility that we want from Rocket Framework projects. Hardcoding data interfaces can easily lead to extra costs later.

Rocket Framework can use a parameter filter and sorting system. The basic idea is that we have settings that are SQL filter code, with tokens that can be replaced with the required data.

# The SQL Settings

The SQL settings code are simply SQL code that has tokens, the best was to explain is to have an example…

*Example Filter*

and R1.[XMLData].value('(genxml/lang/genxml/textbox/title)[1]','nvarchar(max)') like '%{searchtext}%'

The example above shows a simple filter that is using XML to filter on the title field. It uses a replacement token, that is identified by “{” and “}”. ('%{searchtext}%').

Example: *(This can be done in other ways.)*

public string GetFilterProductSQL(SimplisityInfo paramInfo)

{

FastReplacer fr = new FastReplacer("{", "}", false);

fr.Append(SqlFilterProduct);

var tokenList = fr.GetTokenStrings();

foreach (var token in tokenList)

{

var tok = "r/" + token;

if (paramInfo.GetXmlProperty(tok) == "") return ""; // no data so ignore filter

fr.Replace("{" + token + "}", paramInfo.GetXmlProperty(tok));

}

var filtersql = " " + fr.ToString() + " ";

return filtersql;

}

In the above method we use the SessionParams, paramInfo data class. But any method of token replacement may work.

The SQL filter must be designed to fit into the “DNNrocket\_GetList” SPROC, so it is not a full SQL filter with a “WHERE” clause, this is because the SPROC will always have a where clause injected automatically for the EntityTypeCode. This filter statement will be added after that SQL “WHERE” statement, so we start the filter with an “AND” clause.

*Example Order By*

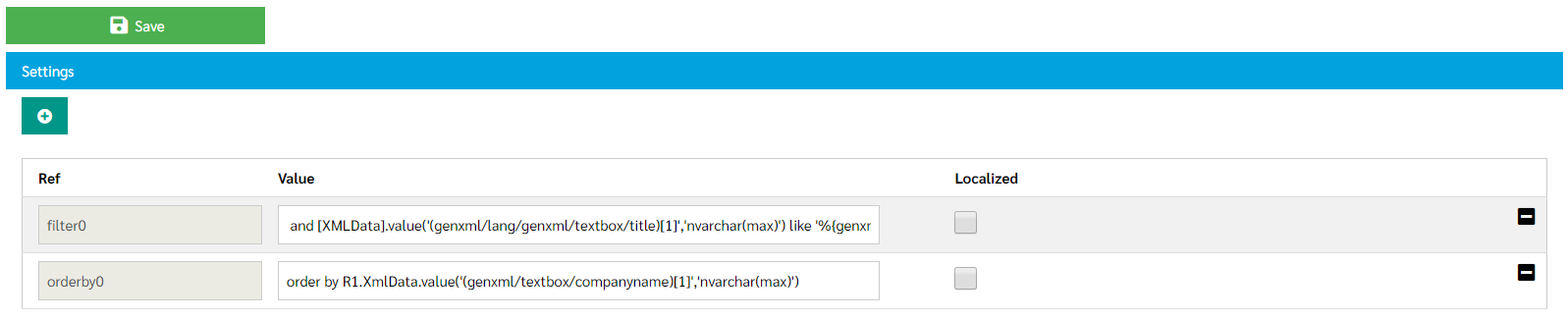
order by R1.XmlData.value('(genxml/textbox/companyname)[1]','nvarchar(max)')

As you can see, the order by SQL is standard SQL without any tokens to be replace.

There can be multiple SQL settings for each type of sort, like is a dropdown list for order selection. The setting key so any selection is a number that matches the appendix of the key.

The order by key in this example is: **orderby#**

Below is an example from RocketMod.



# SQL table in SPROC

The filter and order by settings are designed to work with the “DNNrocket\_GetList” SPROC.

It is this SPROC that does the work, the filter and sort are just SQL text that is added onto the end of the SPROC, in the correct place.

Because we have multiple language record in the system, there are 3 table aliases that can be used in the filter and order by statement **[R1], [RLang1]**. However you only need use:

**[RLangIdx]** = All data table. Holding SimplsityInfo Data, with both localized and non-localized data.

If using the **[XMLData]** column, you DO NOT need a table prefix. The SPROCs that deal with the filter and sort will adject to use RLangIdx, which is the most efficient (Indexed) method.

***REMEMBER IN DIFFERETN SITUATION, TABLE NAMES MAY NOT BE NEEDED OR EVEN INVALID.***

# Example Section of “DNNrocket\_GetList” SPROC

----------------------------------------------- DO NON-PAGING

set @STMT = ' SELECT '

if @ReturnLimit > 0

begin

set @STMT = @STMT + ' top ' + convert(nvarchar(10),@ReturnLimit)

end

set @STMT = @STMT + @rtnFields

set @STMT = @STMT + ' FROM ' + @TableName + ' as R1 '

if NOT(@Lang = '')

begin

set @STMT = @STMT + ' left join ' + @TableName + ' as RLang1 on RLang1.ParentItemId = R1.ItemId and RLang1.[Lang] = ''' + @Lang + ''' and RLang1.TypeCode = R1.TypeCode + ''LANG'' '

set @STMT = @STMT + ' left join ' + @TableName + ' as [RLangIdx] on [RLangIdx].ParentItemId = R1.ItemId and [RLangIdx].[Lang] = ''' + @Lang + ''' and [RLangIdx].TypeCode = R1.TypeCode + ''LANGIDX'' '

end

set @STMT = @STMT + @JoinIndex

IF (RIGHT(@TypeCode,1) = '%')

BEGIN

set @STMT = @STMT + 'WHERE R1.TypeCode Like ''' + @TypeCode + ''' ' + @Filter

END ELSE

BEGIN

IF (@TypeCode = '')

BEGIN

set @STMT = @STMT + 'WHERE R1.TypeCode != ''''' + @Filter

END ELSE

BEGIN

set @STMT = @STMT + 'WHERE R1.TypeCode = ''' + @TypeCode + ''' ' + @Filter

END

END

set @STMT = @STMT + ' ' + @OrderBy ----------------------------------------------- END

*Notice where the “@filter” and “@orderby” parameters are added to the SQL.*

# Defining SQL in settings.cshtml

In RocketMod we sometimes define the filter and order by in the settings.cshtml. So we can ensure we have a the default values in the settings.

@{

var settingsData = (SettingsData)Model.List.First();

var info = settingsData.Info;

var filter0 = settingsData.GetValue("filter0");

if (filter0 == "")

{

filter0 = "and [XMLData].value('(genxml/lang/genxml/textbox/title)[1]','nvarchar(max)') like '%{genxml/textbox/searchtext}%' ";

settingsData.SetValue("filter0", filter0);

settingsData.Update();

}

var orderby0 = settingsData.GetValue("orderby0");

if (orderby0 == "")

{

orderby0 = "order by [XmlData].value('(genxml/textbox/companyname)[1]','nvarchar(max)')";

settingsData.SetValue("orderby0", orderby0);

settingsData.Update();

}

}

This could also be done in compiled code or on install.