

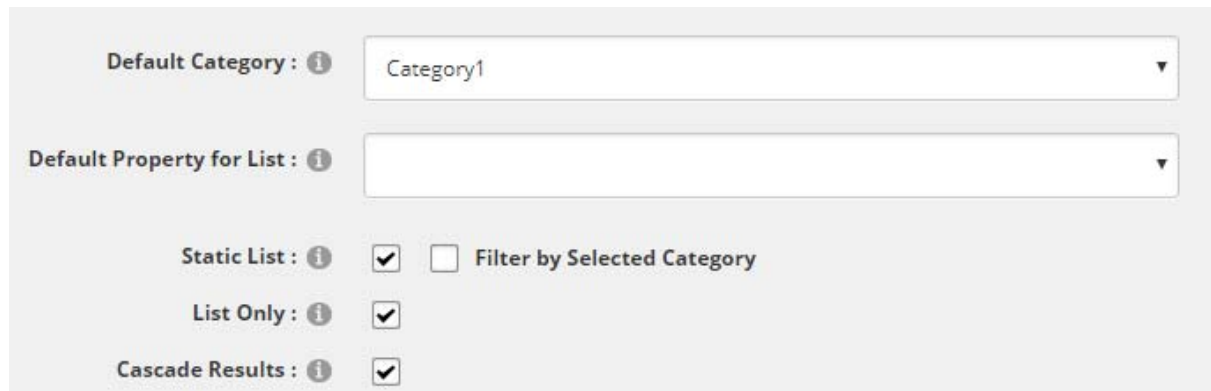
SQL Filter for Product List

Introduction

If you want special filters for product list you can specify the required SQL select as a “@AddPreProcessMetaData” razor token in the template.

Normal Category or Property Selection

In the product module settings you can select a default category to use. You can also decide to have a static list, so when categories are selected, the list does not change.

The screenshot shows a settings panel for a product module. It contains several configuration options: 'Default Category' is set to 'Category1' in a dropdown menu; 'Default Property for List' is an empty dropdown menu; 'Static List' is checked with a checkbox, and 'Filter by Selected Category' is unchecked; 'List Only' is checked with a checkbox; and 'Cascade Results' is checked with a checkbox. Each option has an information icon (i) to its left.

The above will display a “static list” of everything in and in sub categories for “Category1”, selecting categories from menus will not change the list, because “static” has been selected.

You can select either a Category or Property with these settings. If you want multiple properties or categories then you need to use a SQL filter in the template.

If you want to select the list by any other field then you can also use SQL filters.

If you select a default category or property the SQL filter will be applied to the default selection. If you leave the default category and property empty then the SQL filter will be applied across the entire database.

Adding a SQL filter

SQL filters are added to the display template. In the ClassicAjax theme the template used would be “ProductDisplayList.cshtml” or “ProductDisplayAjaxList.cshtml”

Adding a SQL filter is done by using a razor token called: “@AddPreProcessMetaData”.

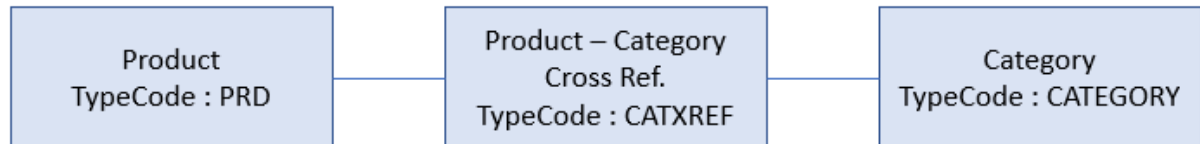
This token not strictly “MetaData”, what it does is pass parameters into cache, which the module can use to process. This token is used where we need to have the data available before the actual razor template is rendered, like for SQL selection.

We’ll just give a few examples to explain how it works, remember if testing it is best to ensure the module settings have no default property or default category selected.

Select Multiple Categories

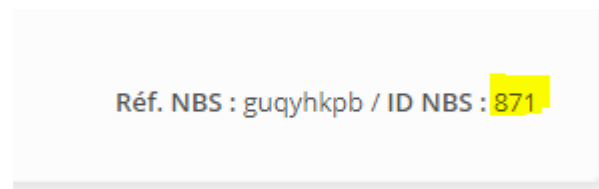
```
@AddPreProcessMetaData("sqlfilter", " and NB1.ItemId in ( select ParentItemId from [NBrightBuy] where TypeCode = 'CATXREF' and (XrefItemId = 873 or XrefItemId = 873)) ", Model.FullTemplateName, Model.ModuleId.ToString())
```

The OpenStore data files are build is such a way the relationship of products and categories is down by using a “CATXREF” TypeCode record.



This is the reason for the sub-select in the SQL to match multiple categories.

You can find the ID of a category by editing the category in the Admin Panel.



Select by product by field value.

```
@AddPreProcessMetaData("sqlfilter", "and nb1.Xmlldata.value('(genxml/textbox/txtproductref)[1] ','nvarchar(max)') = 'TE'", Model.FullTemplateName, Model.ModuleId.ToString())
```

A list of all products fields can be found in the SQL table or if you save a product when in debug mode, a file called “debug_entry.xml” will be saved to the portal root. You simply use the XPATH of the field you need.

Testing Your SQL

To build your SQL for the product list can be difficult. To help you can use the below SQL to test your SQL. Use in SSMS and change the "" value and other values to meet your need.

```
set @PortalId = 0
set @ModuleId = -1
set @Filter = ' and nb1.Xmldata.value('(genxml/textbox/txtproductref)[1] ','nvarchar(max)') = 'TE'
set @OrderBy = ''
set @ReturnLimit = 0
set @TypeCode = 'PRD'
set @Lang = 'fr-FR'
```

NOTE: In the @filter value we use a double quote in this test to escape the quote.

```
SET NOCOUNT ON
DECLARE
    @STMT nvarchar(max)          -- SQL to execute
    ,@rtnFields nvarchar(max)
    ,@NB4cascade nvarchar(max)
    ,@PortalId int
    ,@ModuleId int
    ,@Filter nvarchar(max)
    ,@OrderBy nvarchar(max)
    ,@ReturnLimit int
    ,@TypeCode nvarchar(max)
    ,@Lang nvarchar(max)

set @PortalId = 0
set @ModuleId = -1
set @Filter = ' and nb1.Xmldata.value('(genxml/textbox/txtproductref)[1]
'', 'nvarchar(max)') = 'TE'
set @OrderBy = ''
set @ReturnLimit = 0
set @TypeCode = 'PRD'
set @Lang = 'fr-FR'

IF (@PortalId >= 0) BEGIN

    IF (@ModuleId >= 0) BEGIN
        SET @Filter = ' and (NB1.PortalId = '' + Convert(nvarchar(10),@PortalId) +
        '' or NB1.PortalId = ''-1'') and (NB1.ModuleId = '' + Convert(nvarchar(10),@ModuleId) + '' or
        NB1.ModuleId = ''-1'') ' + @Filter
    END ELSE BEGIN
        SET @Filter = ' and (NB1.PortalId = '' + Convert(nvarchar(10),@PortalId) +
        '' or NB1.PortalId = ''-1'') ' + @Filter
    END

END

SET @Filter = REPLACE(@Filter, '[XMLData]', ' ISNULL(NB2.[XMLData],NB1.[XMLData])')
SET @OrderBy = REPLACE(@OrderBy, '[XMLData]', ' ISNULL(NB2.[XMLData],NB1.[XMLData])')

set @rtnFields = ' NB1.[ItemId] '
set @rtnFields = @rtnFields + ', ISNULL(NB2.[XMLData],NB1.[XMLData]) as [XMLData] '

set @rtnFields = @rtnFields + ', ISNULL(NB2.[Lang], ISNULL(NB1.[Lang], ''')) as [Lang] '

set @rtnFields = @rtnFields + ', NB1.[PortalId] '
set @rtnFields = @rtnFields + ', NB1.[ModuleId] '
set @rtnFields = @rtnFields + ', NB1.[TypeCode] '
set @rtnFields = @rtnFields + ', NB1.[GUIDKey] '
set @rtnFields = @rtnFields + ', NB1.[ModifiedDate] '
set @rtnFields = @rtnFields + ', NB1.[TextData] '
set @rtnFields = @rtnFields + ', NB1.[XrefItemId] '
set @rtnFields = @rtnFields + ', NB1.[ParentItemId] '
set @rtnFields = @rtnFields + ', NB1.[UserId] '

-- Return records without paging.
set @STMT = ' SELECT '
```

```

if @ReturnLimit > 0
begin
    set @STMT = @STMT + ' top ' + convert(nvarchar(10),@ReturnLimit)
end

set @STMT = @STMT + @rtnFields + ' FROM dbo.[NBrightBuy] as NB1 '

set @STMT = @STMT + ' left join  dbo.[NBrightBuyIdx] as NB3 on NB3.ItemId = NB1.ItemId and
NB3.[Lang] = ''' + @Lang + ''''

set @STMT = @STMT + ' left join dbo.[NBrightBuyLang] as NB2 on NB2.ParentItemId = NB1.ItemId
and NB2.[Lang] = ''' + @Lang + ''''

IF (@OrderBy like '%{bycategoryproduct}%')
BEGIN
    DECLARE @categoryid nvarchar(max)
    SET @categoryid = LTRIM(RTRIM(replace(@OrderBy , '{bycategoryproduct}', '')))
    if (@categoryid != '')
    BEGIN
        SET @NB4cascade = ''
        IF CHARINDEX('CATCASCADE',@filter) > 0 SET @NB4cascade = 'or
NB4.TypeCode = 'CATCASCADE'''

        SET @OrderBy = ' order by
NB4.[XMLdata].value('(genxml/sort)[1]','int'), NB3.productname '
        set @STMT = @STMT + ' left join dbo.[NBrightBuy] as NB4 on
(NB4.TypeCode = 'CATXREF' + @NB4cascade + ' ) and NB4.ParentItemId = NB1.ItemId and NB4.XrefItemId
= ' + @categoryid + ''
        END ELSE
        BEGIN
            SET @OrderBy = ' order by NB3.productname '
        END
    END

    IF (RIGHT(@TypeCode,1) = '%')
    BEGIN
        set @STMT = @STMT + ' WHERE NB1.TypeCode Like ''' + @TypeCode + ''' + @Filter + ' '
+ @OrderBy
        END ELSE
        BEGIN
            IF (@TypeCode = '')
            BEGIN
                set @STMT = @STMT + ' WHERE NB1.TypeCode != '''' + @Filter + @OrderBy
            END ELSE
            BEGIN
                set @STMT = @STMT + ' WHERE NB1.TypeCode = ''' + @TypeCode + ''' + @Filter
+ ' ' + @OrderBy
            END
        END

        print @STMT

        EXEC sp_executeSQL @STMT
    
```