

Advanced Form Data Binder

- Introduction
- Advanced Form Data Binder Properties
 - Configure Advanced Form Data Binder
 - Advanced
 - Filter
 - Aggregate Query
 - Expression Columns

Introduction

Advanced Form Row Data Binder is an extended version of the default [Form Data Binder](#). It allows you to add in **Filter Conditions** in a guided and friendly manner.

Advanced Form Data Binder Properties

Configure Advanced Form Data Binder

FORM DATA TABLE NAME *	FIELD	JOIN FIELD ID *
j_expense_claim	j_expense_claim.id	claim

Figure 1: Configure Advanced Form Data Binder

Name	Description								
Form	Source form to retrieve data from.								
Joins Form Data Table	<table border="1"><thead><tr><th>Name</th><th>Description</th></tr></thead><tbody><tr><td>Form Data Table Name</td><td>Target table to join with</td></tr><tr><td>Field</td><td>Target table field to join with</td></tr><tr><td>Join Field Id</td><td>Parent field Id to join with</td></tr></tbody></table>	Name	Description	Form Data Table Name	Target table to join with	Field	Target table field to join with	Join Field Id	Parent field Id to join with
	Name	Description							
	Form Data Table Name	Target table to join with							
	Field	Target table field to join with							
Join Field Id	Parent field Id to join with								



Sample

In the screenshot example in Figure 1, such configurations can be presented with the following SQL.

Sample SQL

```
SELECT * FROM "Claim Entry" entry JOIN hr_expense_claim claim ON claim.id = entry.claim
```

Advanced

Filter

Filter

Filter Conditions

JOIN TYPE *	FIELD *	OPERATOR *	VALUE	
And ▼	category × ▼	Equal ▼	Meal	⏪ ⏩ ✖
Or ▼	category × ▼	Equal ▼	Allw	⏪ ⏩ ✖

+ +

Extra Conditions ? /

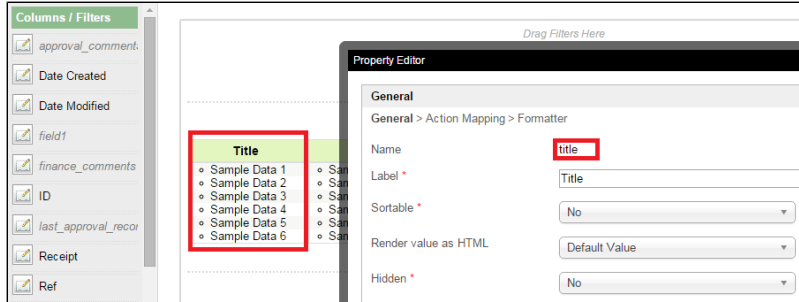
Figure 2: Advanced > Filter

Name	Description				
Filter Conditions	Filter Conditions				
Join Type	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 15%;">Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Join Type</td> <td> <ul style="list-style-type: none"> • And • Or </td> </tr> </tbody> </table>	Name	Description	Join Type	<ul style="list-style-type: none"> • And • Or
Name	Description				
Join Type	<ul style="list-style-type: none"> • And • Or 				
Field	<p>Field ID. (e.g. title)</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 5px;"> <div style="display: flex; align-items: flex-start;"> <div style="width: 25%; border-right: 1px solid gray; padding-right: 5px;"> <p style="font-size: small; margin: 0;">Columns / Filters</p> <ul style="list-style-type: none"> <input type="checkbox"/> approval_comment <input type="checkbox"/> Date Created <input type="checkbox"/> Date Modified <input type="checkbox"/> field1 <input type="checkbox"/> finance_comments <input type="checkbox"/> ID <input type="checkbox"/> last_approval_recoi <input type="checkbox"/> Receipt <input type="checkbox"/> Ref </div> <div style="width: 75%; padding-left: 5px;"> <p style="font-size: x-small; margin: 0; text-align: center;">Drag Filters Here</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 5px;"> <p style="font-size: x-small; margin: 0;">Property Editor</p> <p style="margin: 0;">General</p> <p style="font-size: x-small; margin: 0;">General > Action Mapping > Formatter</p> <p style="margin: 0;">Name title</p> <p style="margin: 0;">Label * <input type="text" value="Title"/></p> <p style="margin: 0;">Sortable * <input type="text" value="No"/></p> <p style="margin: 0;">Render value as HTML <input type="text" value="Default Value"/></p> <p style="margin: 0;">Hidden * <input type="text" value="No"/></p> </div> </div> </div> </div>				
Operator	<ul style="list-style-type: none"> • Equal • Not Equal • Greater Than • Greater Than Or Equal • Less Than • Less Than Or Equal • Like • Not Like • In • Not In • Is True • Is False • Is Null • Is Not Null 				
Value	Filter value				

Extra Conditions Additional condition(s) for filtering the data set. HQL is expected here.

Syntax Query

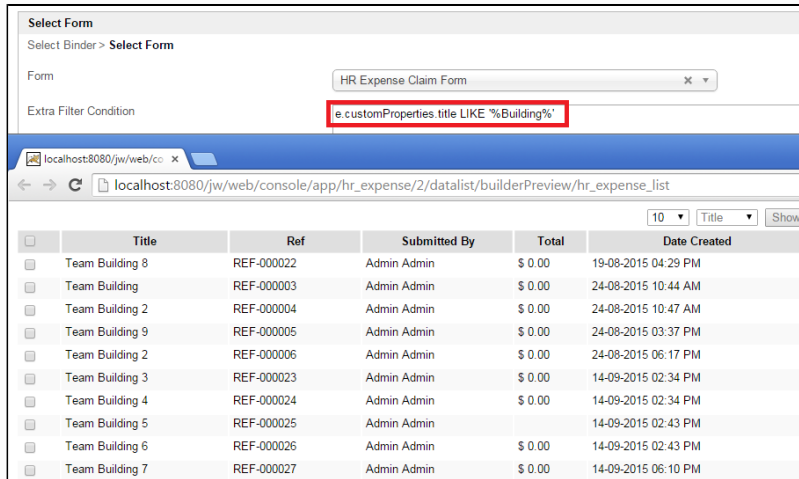
Start your filter name with `e.customProperties`, followed by the field id (i.e. `title`)



The screenshot shows a 'Property Editor' dialog box. The 'General' tab is active, and the 'Name' field is set to 'title'. The 'Label' field is set to 'Title'. The 'Sortable' field is set to 'No'. The 'Render value as HTML' field is set to 'Default Value'. The 'Hidden' field is set to 'No'. In the background, a list of sample data items is visible, with the 'Title' column highlighted in red.

HQL is accepted

You may even use an operator such as "LIKE" to narrow down your data set.



The screenshot shows a 'Select Form' dialog box. The 'Form' field is set to 'HR Expense Claim Form'. The 'Extra Filter Condition' field is set to 'e.customProperties.title LIKE '%Building%''. Below the dialog is a browser window showing a table of expense claims.

	Title	Ref	Submitted By	Total	Date Created
<input type="checkbox"/>	Team Building 8	REF-000022	Admin Admin	\$ 0.00	19-08-2015 04:29 PM
<input type="checkbox"/>	Team Building	REF-000003	Admin Admin	\$ 0.00	24-08-2015 10:44 AM
<input type="checkbox"/>	Team Building 2	REF-000004	Admin Admin	\$ 0.00	24-08-2015 10:47 AM
<input type="checkbox"/>	Team Building 9	REF-000005	Admin Admin	\$ 0.00	24-08-2015 03:37 PM
<input type="checkbox"/>	Team Building 2	REF-000006	Admin Admin	\$ 0.00	24-08-2015 06:17 PM
<input type="checkbox"/>	Team Building 3	REF-000023	Admin Admin	\$ 0.00	14-09-2015 02:34 PM
<input type="checkbox"/>	Team Building 4	REF-000024	Admin Admin	\$ 0.00	14-09-2015 02:34 PM
<input type="checkbox"/>	Team Building 5	REF-000025	Admin Admin	\$ 0.00	14-09-2015 02:43 PM
<input type="checkbox"/>	Team Building 6	REF-000026	Admin Admin	\$ 0.00	14-09-2015 02:43 PM
<input type="checkbox"/>	Team Building 7	REF-000027	Admin Admin	\$ 0.00	14-09-2015 06:10 PM

Sample

```
e.customProperties.title = 'Trip'
```

A hash variable is accepted here.

Sample

```
e.customProperties.submitted_by = '#currentUser.id#'
```

Userview Key can be used as part of the condition.

Sample

```
e.customProperties.category_id = '#userviewKey#'
```

Aggregate Query

Aggregate Query

Group By ?

- j_expense_claim.CreatedDate
- j_expense_claim.FinanceApprovedBy
- j_expense_claim.FinanceApprovedDate
- j_expense_claim.SelectApprover
- j_expense_claim.approval_comments
- j_expense_claim.createdBy
- j_expense_claim.createdByName
- j_expense_claim.dateCreated

Aggregate Fields

FIELD *	FUNCTION *
amount	Sum
j_expense_claim.title	Count

Having Conditions

JOIN TYPE *	FIELD *	FUNCTION *	OPERATOR *	VALUE
And	j_expense_claim.total	Sum	Greater Than	1000

Figure 3: Advanced > Aggregate Query

DATALIST BUILDER Expenses Claim v3: Expense Claim Entry (Published)

SOURCE DESIGN PROPERTIES PREVIEW SAVE

Columns / Filters

- Amount (SUM)
- j_expense_claim.claimant
- j_expense_claim.title (COUNT)

Actions

- Bean Shell
- Hyperlink
- Delete
- JDBC

Drag Filters Here

Drag Columns Here

j_expense_claim.title (COUNT)	Amount (SUM)	j_expense_claim.claimant
AFD_COUNT_j_expense_claim.title	AFD_SUM_amount	j_expense_claim.claimant
Unsortable	Unsortable	Unsortable
Visible	Visible	Visible
Exportable	Exportable	Exportable
Default	Default	Default
-	-	-
-	-	-

<input type="checkbox"/>	j_expense_claim.title (COUNT)	Amount (SUM)	j_expense_claim.claimant
<input type="checkbox"/>	2	1600	Admin Admin
<input type="checkbox"/>	1	8000	Cat Grant

2 items found, displaying all items.

1

CSV | Excel | XML | PDF

Figure 4: The configurations shown in Figure 3 will produce the following sample result.

Name	Description
Group By	Add grouping clause/function to the eventual data set. This can be used together with Aggregate Fields above. In figure 3, the "amount" field will be summed up by "claimant", shown in per record row.

Aggregate Fields	<p>This field will be displayed once any number of columns has been added into the Group By field.</p> <p>The select field is to aggregate.</p> <ul style="list-style-type: none"> • Count • Count Distinct • Sum • Min • Max • Avg <p>In the sample screenshot above, the "amount" field will be put into the "Sum" function, and "Count" will be applied to "title".</p>
Having Conditions	<p>This field will be displayed once any number of columns has been added into the Group By field.</p> <p>The HAVING clause enables you to specify conditions that filter which group results appear in the final results. The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause. Read more at http://www.dofactory.com/sql/having</p>

Expression Columns

Expression Columns

Expression Columns ?

ALIAS *	EXPRESSION *
price	CAST(price AS long)

+ - +

Custom Checkbox/Radio Button Value ?

Figure 5: Advanced > Expression Columns

Name	Description
Expression Columns	<p>An additional column can be added in this expression columns using Hibernate Query Language (HQL). This is especially useful when you need to perform additional computation on multiple columns.</p> <p>Example 1 - Cast column to data type "long".</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p>Expression</p> <pre>CAST(price AS long)</pre> </div> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p>Expression</p> <pre>CAST(e.customProperties.sales_price AS long) - CAST(e.customProperties.price AS long)</pre> </div> <p>Example 2 - Concatenate multiple columns into one.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p>Expression</p> <pre>CONCAT(first_name, ' ', last_name)</pre> </div> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Expression</p> <pre>first_name ' ' last_name</pre> </div>

Custom Checkbox/Radio Button
Value

Define custom record ID to be used to pass over to column action. Defaulted to ID.

All Expenses History

Delete

<input type="checkbox"/>	#	TITLE	\$	STATUS	SUBMIT BY
<input checked="" type="checkbox"/>	0001	Claim A	111.00	Approved	Admin Admin