

How to configure a TrendConfigurator control whose tags come from a database

The TrendConfigurator control provides users the ability to configure and save trended values in a set, which they can then reload when they next logon. In this way, each user can customize the values trended using an associated LineChart control, according to their requirements.

Note: The TrendConfigurator is a SEPARATE control from the LineChart control and requires an associated LineChart control to trend its configured sets of series.

Since this control is intended for use in a SCADA environment, in order for users to specify which values (tags) they need to trend, the following combo boxes are used:

1. **Plant Area:** this is the **Area** level of physical structure of automation projects described by the ISA S95 and S88 standards. The level above this is the Site level. The Area contains one or more Process Cells or PLCs, which in Adroit-speak are communicated to via Devices, hence the next tag-locating combo box is:
2. **Device:** this is the **Process Cell** level of physical structure of automation projects described by the ISA S95 and S88 standards. Which in SCADA projects equate to the PLCs of each defined plant Area. This is called Device, since Adroit uses a Device to communicate to each PLC, within a Plant Area. Each Device contains one or more values, hence the final tag-locating combo box is:
3. **Pen Description:** this is the **Unit** and **Equipment module** and the **Control module** levels of the physical structure of automation projects described by the ISA S95 and S88 standards. This contains all the values (tags) within the specified Device.

This document describes how to connect a TrendConfigurator control to a LineChart control for runtime chart configuration, when the physical structure of its tags (adhering to the ISA S95 and S88 standards) are sourced from a database.

Note: This document assumes that the reader has a basic working knowledge of using SQL databases and using databases in Smart UI.

This document describes the following steps that are required:

1. [Add an Adroit Datasource that connects to the Agent Server containing the tags to trend.](#)
2. [Add an OleDb datasource and connect it to the database where the structured list of the areas, devices and their tags are stored.](#)
3. [Create the following three Select queries for this OleDb datasource:](#)
 - [Areas Select query](#)
 - [DevicesPerArea Select query](#)
 - [TagsPerDevice Select query](#)
4. [Add a LineChart control to trend the tags.](#)
5. [Add a TrendConfigurator control to perform the runtime configuration of the LineChart.](#)
6. [Link the three TrendConfigurator combo boxes to their respective Select queries:](#)
 - [Add and configure a Query spider for the Areas Select query](#)
 - [Troubleshoot query and link the Return Object output to the AreaList property](#)
 - [Add and configure a Query spider for the DevicesPerArea Select query](#)
 - [Link the CurrentArea property to supply the value of its area \(parameter\) input](#)
 - [Troubleshoot query and link the Return Object output to the DevicesList property](#)
 - [Add and configure a Query spider for the TagsPerDevice Select query](#)
 - [Link the CurrentArea property to supply the value of its area \(parameter\) input](#)
 - [Link the CurrentDevice property to supply the value of the device \(parameter\) input](#)
 - [Troubleshoot query and link the Return Object output to the PenList property](#)
7. [Operate the TrendConfigurator control at runtime.](#)

[Back to top](#)

Adding an Adroit datasource

Create an Adroit datasource and name it **Adroit** and specify the Agent Server or clustered Agent Servers that contain the tags that you need to trend. For details, see [Adding Adroit datasources](#).

[Back to top](#)

Adding an OleDB datasource

Create an OleDB datasource and name it **TrendTags** and link it to the database containing the table containing the plant areas, their devices and the Adroit tags that your operators need to trend, in this example this table is called **Tags**. For details, see [Adding OLE DB datasources](#).

The structure of the **Tags** database table:

	ID	Area	Device	Tag
▶	1	PLTA	PLCA	PLTA_PLCA_FLOW01.value
	2	PLTA	PLCA	PLTA_PLCA_PRESS01.value
	3	PLTA	PLCA	PLTA_PLCA_LEVEL01.value
	4	PLTA	PLCB	PLTA_PLCB_FLOW02.value
	5	PLTA	PLCB	PLTA_PLCB_PRESS02.value
	6	PLTA	PLCB	PLTA_PLCB_LEVEL02.value
	7	PLTB	PLCC	PLTB_PLCC_FLOW03.value
	8	PLTB	PLCC	PLTB_PLCC_PRESS03.value
	9	PLTB	PLCC	PLTB_PLCC_LEVEL03.value
	10	PLTB	PLCD	PLTB_PLCD_FLOW04.value
	11	PLTB	PLCD	PLTB_PLCD_PRESS04.value
	12	PLTB	PLCD	PLTB_PLCD_LEVEL01.value
*	NULL	NULL	NULL	NULL

[Back to top](#)

Add three Select queries to this OleDB datasource

The TrendConfigurator provides three combo box fields, per configured series or tag, namely **Area**, **Device** and **Pen** which can be driven by using their respective properties **AreaList**, **DeviceList** and **PenList** - either via Select queries on an OleDB datasource or via direct links.

Note: Driving both the **Area** and **Device** combo boxes are optional, if you do not wish to select a list of tags per device per area and just want to drive the **Pen** column combo boxes with a list of all tags instead.

If ONLY driving the **Pen** combo box, then the required syntax of the **Select Query** that populates this combo box, is as follows:

SELECT DISTINCT [Table].[Tags] FROM [Table], assuming that the field of **Table** containing the tags is called **Tags**.

In this example, we will create three Select queries namely Areas, DevicesPerArea and TagsPerDevice to populate each combo box, as follows:

[Back to top](#)

Areas Select query

1. Add a **Select Query** to this **TrendTags** OleDB datasource, called **Areas**, which returns a unique list of areas from the **Area** field of the **Tags** database table. For details, see [Select queries](#).
2. Ensure that the resultant query adheres to the following syntax:
SELECT DISTINCT [Tags].[Area] FROM [Tags]
3. Click **Preview** to test that this query works as intended.

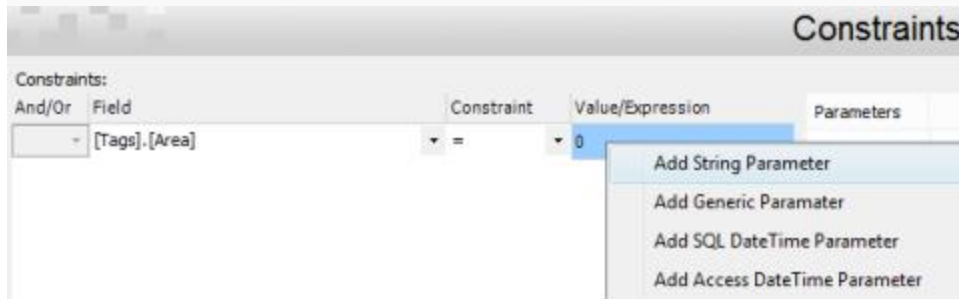
- Click **Finish** to add this query to the **Select Queries** queries category of this **TrendTags** OleDb datasource.

[Back to top](#)

DevicesPerArea Select query

- Add a **Select Query** to this **TrendTags** OleDb datasource, called **DevicesPerArea**, which returns a unique list of devices from the **Device** field of the **Tags** database table for a specific area.
Since this query is more complex than the previous one, as it requires a constraint that uses a String parameter, use the query builder option instead of the Custom query option, in this case after selecting the **Device** field of the **Tags** database table:
- Click **Next** until you get to the **Constraints** section of the configuration dialog.
- Select [Tags].[Area] as the **Field**.
- Select = as the **Constraint** operator.
- Right click the **Value/Expression** and select **Add String Parameter**.
- Type in the name "area" for this parameter, to add this to the **Parameters** list.

Example:



- Ensure that the resultant query adheres to the following syntax:
SELECT DISTINCT [Tags].[Device] FROM [Tags] WHERE [Tags].[Area] = '<!!area!!>'
Note: You can simply copy and paste this query as-is into the **Custom Query** dialog, you do not need to specifically add parameters and/or constraints by using the **Select Query Builder** dialogs. The required parameter data type is specified by its specific formatting in the query string, whether this is typed in manually or selected from the right click menu.
- Click **Preview** to test that this query works as intended and then click **Finish**.

[Back to top](#)

TagsPerDevice Select query

- Add a **Select Query** to this **TrendTags** OleDb datasource, called **TagsPerDevice**, which returns a unique list of tags from the **Tag** field of the **Tags** database table for a specific device in a specific area.
Since this query is more complex than the previous one, as it requires a constraint that uses a String parameter, use the query builder option instead of the Custom query option, in this case after selecting the **Tag** field of the **Tags** database table:
- Click **Next** until you get to the **Constraints** section of the configuration dialog.
- Select [Tags].[Area] as the **Field**.
- Select = as the **Constraint** operator.
- Right click the **Value/Expression** and select **Add String Parameter**.
- Type in the name "area" for this parameter, to add this to the **Parameters** list.
- Click the **Add Constraint** button to add another constraint to this list of constraints.
- Select [Tags].[Device] as the **Field**.

9. Select = as the **Constraint** operator.
10. Right click the **Value/Expression** and select **Add String Parameter**.
11. Type in the name "**device**" for this parameter, to add this to the **Parameters** list.

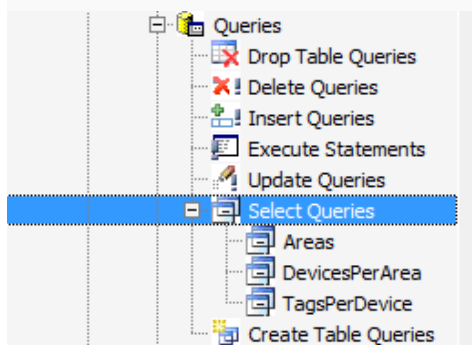
Example:

Constraints:				Parameters
And/Or	Field	Constraint	Value/Expression	
-	[Tags].[Area]	=	<!!area!!>	area
AND	[Tags].[Device]	=	<!!device!!>	device

12. Ensure that the resultant query adheres to the following syntax:
SELECT DISTINCT [Tags].[Tag] FROM [Tags] WHERE [Tags].[Area] = '<!!area!!>' AND [Tags].[Device] = '<!!device!!>'
Note: You can simply copy and paste this query as-is into the **Custom Query** dialog, you do not need to specifically add parameters and/or constraints by using the **Select Query Builder** dialogs. The required parameter data type is specified by its specific formatting in the query string, whether this is typed in manually or selected from the right click menu.
13. Click **Preview** to test that this query works as intended and then click **Finish**.

All three queries should be listed in the **Select Queries** category of the OLEDB datasource.


Example:



[Back to top](#)

Add and configure a LineChart control to chart the tag data

By default the LineChart is configured to chart real-time values, so you need to change its properties as you are charting historical (logged) data from a database.

1. If necessary, add the required project and/or graphic form to this project.
2. Add a **LineChart** control to this graphic form, from the **Data** category of the **Toolbox** window. This displays the **Series Setup** page of the chart configuration dialog.
3. Since the users will only be adding series at run-time, delete the default series by selecting it and clicking the **Delete Series** button, at the bottom of the series list and then click **Finish** to close this chart configuration dialog.
4. Click the  glyph, in the top left corner of the LineChart control, which lists its commonly-used properties and configuration tasks.

In the **Data Settings** section of this task list:

- Uncheck the **Realtime** checkbox, which is checked by default and not required since this chart is trending data from a database and not real-time data

In the **Configuration Dialogs** section of this task list:

- Click the **Edit Time Details...** link to display the **Time Detail Settings** dialog, since we need to ensure that the time interval of this LineChart control is sufficient for the tags that it will be trending, as follows:

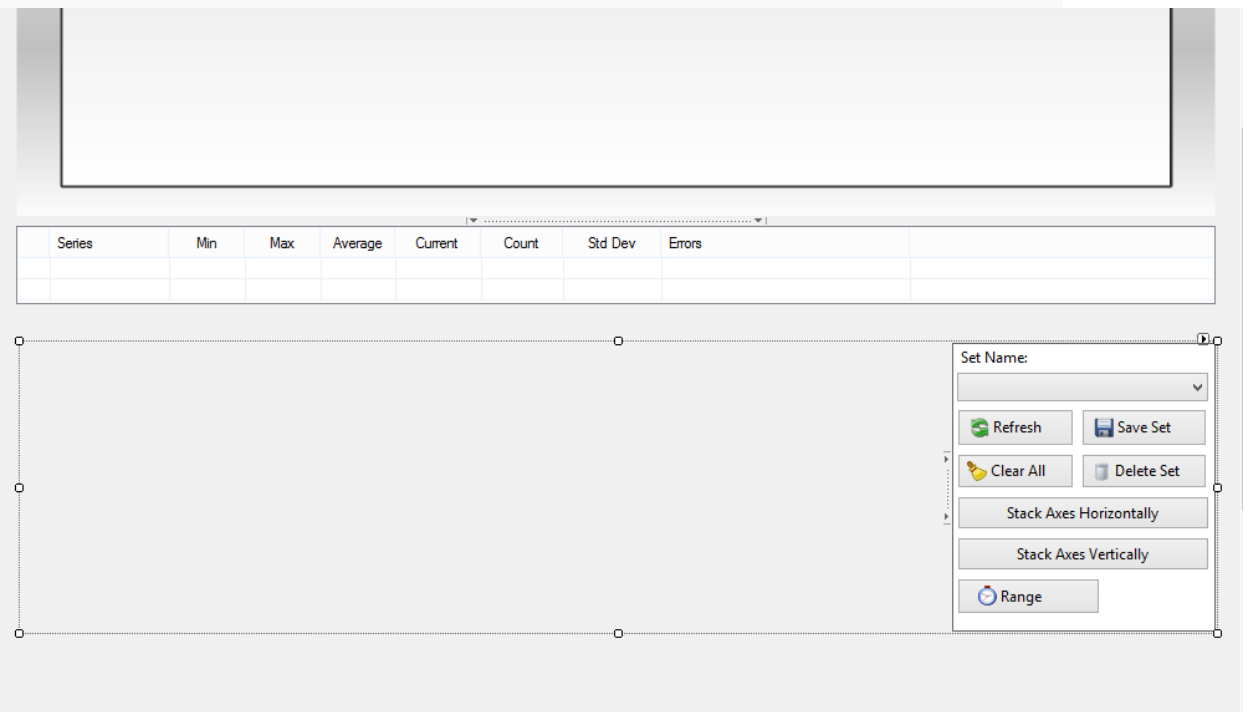
Leave the **Period Type** as **End DateTime - Timespan** and ensure that the specified length of the **Timespan** is correct and that this specifies the correct time unit, by selecting the required time unit from the list box i.e Minutes, Hours, Days etc and then click the **Finish** button.


[Back to top](#)

Add and configure the TrendConfigurator control

1. Add a **TrendConfigurator** control to this graphic form, from the **Data** category of the **Toolbox** window, typically position this control below its associated LineChart control.

Example:



2. Click the  glyph, in the top left corner of the TrendConfigurator control, which lists its commonly-used properties and configuration tasks.
 - Change the **Datasource Mode** setting to **Custom** (by default this is set to MAPS).
 - In the **Adroit DataSource** list, select the name of the Adroit datasource that connects to the Agent Server that contains the tags that you want your users to trend (and, if necessary, the plant areas and devices of these tags too). For this example, this is called **Adroit**.
 - In the **Chart Name** list, select the name of the LineChart control, on this graphic form, that this TrendConfigurator control will configure. For this example, there is only one LineChart control on this graphic form, called lineChart.

[Back to top](#)

Connect the three TrendConfigurator combo boxes to their respective Select queries

1. Click the **Spider Configuration** tab to configure the spider engine of this graphic form, as follows:
2. Right click the spider workspace and select the **Hide Unselected Spiders/Regions** sub-menu and select **None** - so that you can see all the spiders that you add to this spider workspace.

[Back to top](#)

Add and configure a Query spider for the Areas Select query

Note: Query spiders can perform queries and execute statements provided by any OLE DB datasource.

1. Right click the spider workspace and select **Create New Spider** and select the **Query** spider from the **Databasing** category.
2. Double click the spider icon to display its spider configuration dialog.
3. Double click the default name, for instance **Query1** and type in the new name: **Areas**.
4. Configure the **Inputs** section of this spider to specify the **Select Query** created for the **TrendTags** OleDB datasource, which is called **Areas** , as follows:
5. Set the **Server** input to **Default Server**; the **OLEDB Data Source** input to **TrendTags**; and the **Query Type** input to **Select Query** and the **Query Name** input to **Areas**
6. Configure this Query spider to perform this query when the graphic form is opened, by configuring the **Event Trigger** section, to select **Form** as the **Event Object** and **Load** as the **Event Name**.

Troubleshoot query and link the Return Object output to the AreaList property

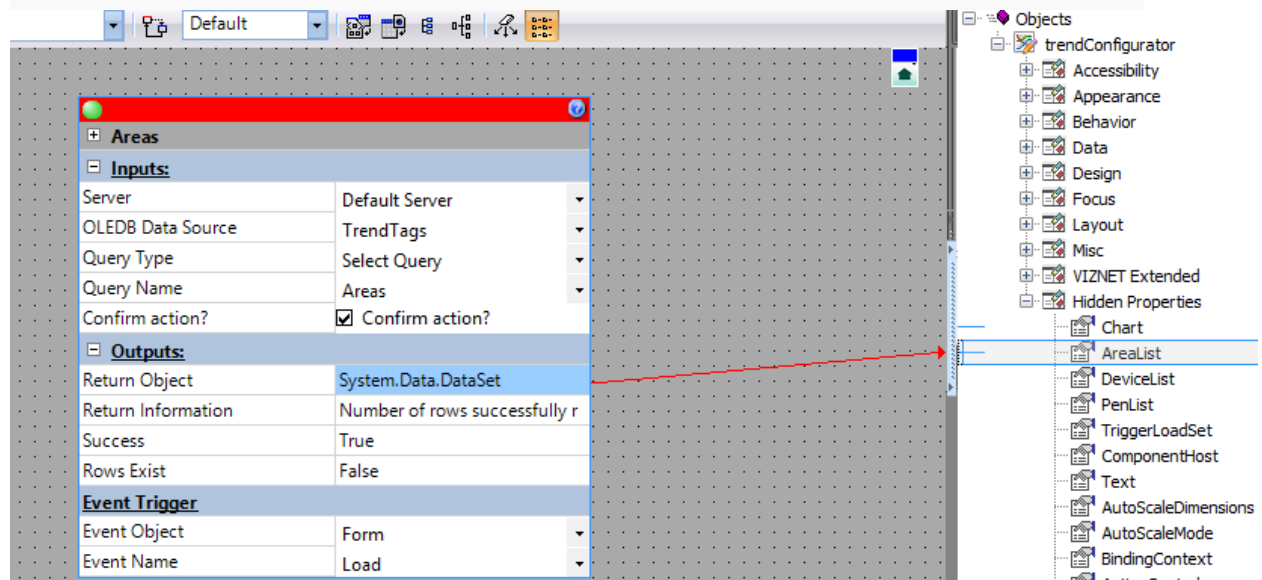
7. Right click the Spider header (bar at the top of this **Areas** spider) and select **Trigger Now!** to get the initial dataset for this Query spider.

Note: Ensure that the **Success** output is True (i.e. the query ran successfully), if this is False then there is either a problem with the query syntax or another problem preventing this query from running.

Tip: The **Rows Exist** output is only True when the query returns data and is False if an empty dataset is returned.

8. In the graphic form, click the **trendConfigurator** control to display this control's properties in the **Objects** tree on the right of the spider workspace.
9. In the **Outputs** section of the **Areas** Query spider, drag the **Return Object** output onto the **Hidden Properties** property group (to expand it) and then onto the **AreaList** property of the **trend-Configurator** control to connect them with a Silk.
10. Click **Finish**, when the **Filter DataSet** configuration dialog is displayed.

Example:



[Back to top](#)

Add and configure a Query spider for the DevicesPerArea Select query

1. Right click the spider workspace and select **Create New Spider** and select the **Query** spider from the **Databasing** category.

2. Double click the spider icon to display its spider configuration dialog.
3. Double click the default name, for instance **Query1** and type in the new name: **DevicesPerArea** .
4. Configure the **Inputs** section of this spider to specify the **Select Query** created for the **TrendTags** OleDb datasource, which is called **DevicesPerArea**, as follows:
5. Set the **Server** input to **Default Server**; the **OleDb Data Source** input to **TrendTags**; and the **Query Type** input to **Select Query** and the **Query Name** input to **DevicesPerArea**
You should see that once this query is selected that the name of its parameter, namely **area** is also added to the **Inputs** section.
6. Right click the Spider header (bar at the top of this **DevicesPerArea** spider) and, if necessary, enable the Silk triggering method (add a tick to the **Enable Silk Triggering** option) and disable Event triggering (remove the tick from the **Trigger Spider From Event** option).

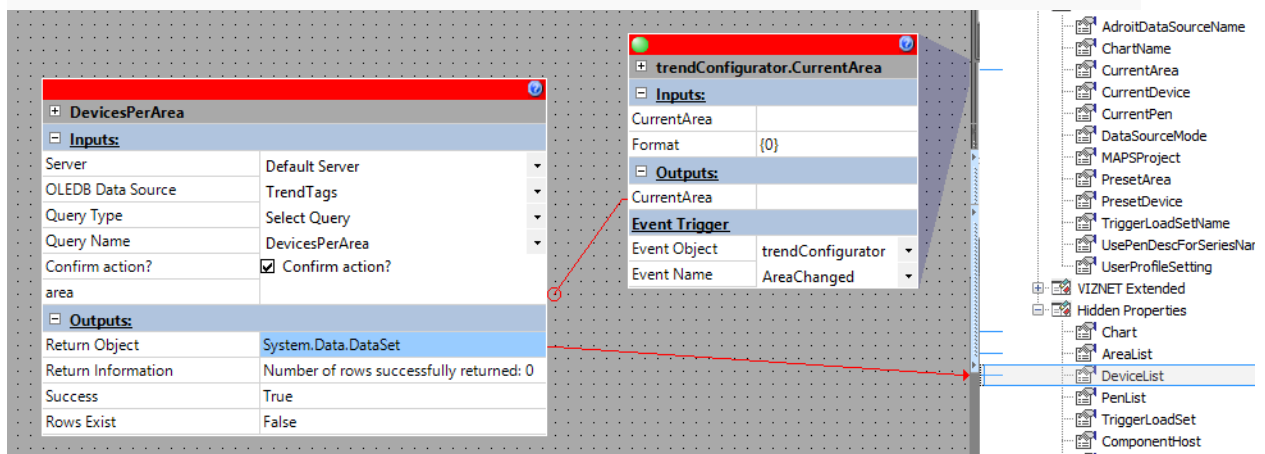
Link the CurrentArea property to supply the value of the area (parameter) input

7. In the graphic form, click the **trendConfigurator** control to display this control's properties in the **Objects** tree on the right of the spider workspace.
8. In the **Objects** tree on the right of the spider workspace, expand the **Misc** property group and drag the **CurrentArea** property to the **area** input of the **DevicesPerArea** spider to connect them with a Silk.
9. Right click the **CurrentArea** property in the **Objects** tree on the right of the spider workspace and select **Show property spider**.
10. Right click the Spider header (bar at the top of this **trendConfigurator.CurrentArea** spider) and, if necessary, disable the default Silk triggering method (remove the tick from the **Enable Silk Triggering** option) and enable Event triggering instead (add a tick to the **Trigger Spider From Event** option)
11. In the **Event Trigger** section of this **trendConfigurator.CurrentArea** spider, configure this **CurrentArea** property of the **trendConfigurator** control to update when its area is changed, by selecting **AreaChanged** from the **Event Name** list, since the **Event Object** should already specify **trendConfigurator**.

Troubleshoot query and link the Return Object output to the DevicesList property

12. Right click the Spider header (bar at the top of this **DevicesPerArea** spider) and select **Trigger Now!** to get the initial dataset for this Query spider.
Note: Ensure that the **Success** output is True (i.e. the query ran successfully), if this is False then there is either a problem with the query syntax or another problem preventing this query from running.
Tip: The **Rows Exist** output is only True when the query returns data and is False if an empty dataset is returned.
13. Right click the Spider header (bar at the top of this **DevicesPerArea** spider) and select **Trigger Now!** to get the initial dataset for this Query spider.
14. In the **Outputs** section of the **DevicesPerArea** Query spider, drag the **Return Object** output onto the **Hidden Properties** property group (to expand it) and then onto the **DeviceList** property of the **trendConfigurator** control to connect them with a Silk.
15. Click **Finish**, when the **Filter DataSet** configuration dialog is displayed.

Example:



[Back to top](#)

Add and configure a Query spider for the TagsPerDevice Select query

1. Right click the spider workspace and select **Create New Spider** and select the **Query** spider from the **Databasing** category.
2. Double click the spider icon to display its spider configuration dialog.
3. Double click the default name, for instance **Query1** and type in the new name: **TagsPerDevice**.
4. Configure the **Inputs** section of this spider to specify the **Select Query** created for the **TrendTags** OleDB datasource, which is called **TagsPerDevice**, as follows:
5. Set the **Server** input to **Default Server**; the **OLEDB Data Source** input to **TrendTags**; and the **Query Type** input to **Select Query** and the **Query Name** input to **TagsPerDevice**
You should see that once this query is selected that the names of its parameters, namely **area** and **device** are also added to the **Inputs** section.
6. Right click the Spider header (bar at the top of this **TagsPerDevice** spider) and, if necessary, enable the Silk triggering method (add a tick to the **Enable Silk Triggering** option) and disable Event triggering (remove the tick from the **Trigger Spider From Event** option).

Link the CurrentArea property to supply the value of the area (parameter) input

7. In the graphic form, click the **trendConfigurator** control to display this control's properties in the **Objects** tree on the right of the spider workspace.
8. In the **Objects** tree on the right of the spider workspace, expand the **Misc** property group and drag the **CurrentArea** property to the **area** input of the **TagsPerDevice** spider to connect them with a Silk.
9. Right click the **CurrentArea** property in the **Objects** tree on the right of the spider workspace and select **Show property spider**.
10. Right click the Spider header (bar at the top of this **trendConfigurator.CurrentArea** spider) and, if necessary, disable the default Silk triggering method (remove the tick from the **Enable Silk Triggering** option) and enable Event triggering instead (add a tick to the **Trigger Spider From Event** option)
11. In the **Event Trigger** section of this **trendConfigurator.CurrentArea** spider, configure this **CurrentArea** property of the **trendConfigurator** control to update when its area is changed, by selecting **AreaChanged** from the **Event Name** list, since the **Event Object** should already specify **trendConfigurator**.

Link the CurrentDevice property to supply the value of the device (parameter) input

12. In the **Objects** tree on the right of the spider workspace, expand the **Misc** property group and drag the **CurrentDevice** property to the **device** input of the **TagsPerDevice** spider to connect them with a Silk.

13. Right click the **CurrentDevice** property in the **Objects** tree on the right of the spider workspace and select **Show property spider**.
14. Right click the Spider header (bar at the top of this **trendConfigurator.CurrentDevice** spider) and, if necessary, disable the default Silk triggering method (remove the tick from the **Enable Silk Triggering** option) and enable Event triggering instead (add a tick to the **Trigger Spider From Event** option)
15. In the **Event Trigger** section of this **trendConfigurator.CurrentDevice** spider, configure this **CurrentDevice** property of the **trendConfigurator** control to update when its device is changed, by selecting **DeviceChanged** from the **Event Name** list, since the **Event Object** should already specify **trendConfigurator**.
16. In the **Objects** tree on the right of the spider workspace, expand the **Misc** property group and drag the **CurrentDevice** property to the **device** input of the **TagsPerDevice** spider to connect them with a Silk.

Troubleshoot query and link the Return Object output to the PenList property

17. Right click the Spider header (bar at the top of this **TagsPerDevice** spider) and select **Trigger Now!** to get the initial dataset for this Query spider.

Note: Ensure that the **Success** output is True (i.e. the query ran successfully), if this is False then there is either a problem with the query syntax or another problem preventing this query from running.

Tip: The **Rows Exist** output is only True when the query returns data and is False if an empty dataset is returned.
18. In the **Outputs** section of the **TagsPerDevice** Query spider, drag the **Return Object** output onto the **Hidden Properties** property group (to expand it) and then onto the **PenList** property of the **trendConfigurator** control to connect them with a Silk.
19. Click **Finish**, when the **Filter DataSet** configuration dialog is displayed.

Example:

The screenshot displays the Adroit Technologies spider workspace. On the left, the **TagsPerDevice** spider configuration is shown with the following properties:

TagsPerDevice	
Inputs:	
Server	Default Server
OleDb Data Source	TrendTags
Query Type	Select Query
Query Name	TagsPerDevice
Confirm action?	<input checked="" type="checkbox"/> Confirm action?
area	
device	
Outputs:	
Return Object	System.Data.DataSet
Return Information	Number of rows successfully returned: 0
Success	True
Rows Exist	False

In the center, the **trendConfigurator.CurrentArea** spider configuration is shown:

trendConfigurator.CurrentArea	
Inputs:	
CurrentArea	
Format	{0}
Outputs:	
CurrentArea	
Event Trigger	
Event Object	trendConfigurator
Event Name	AreaChanged

Below it, the **trendConfigurator.CurrentDevice** spider configuration is shown:

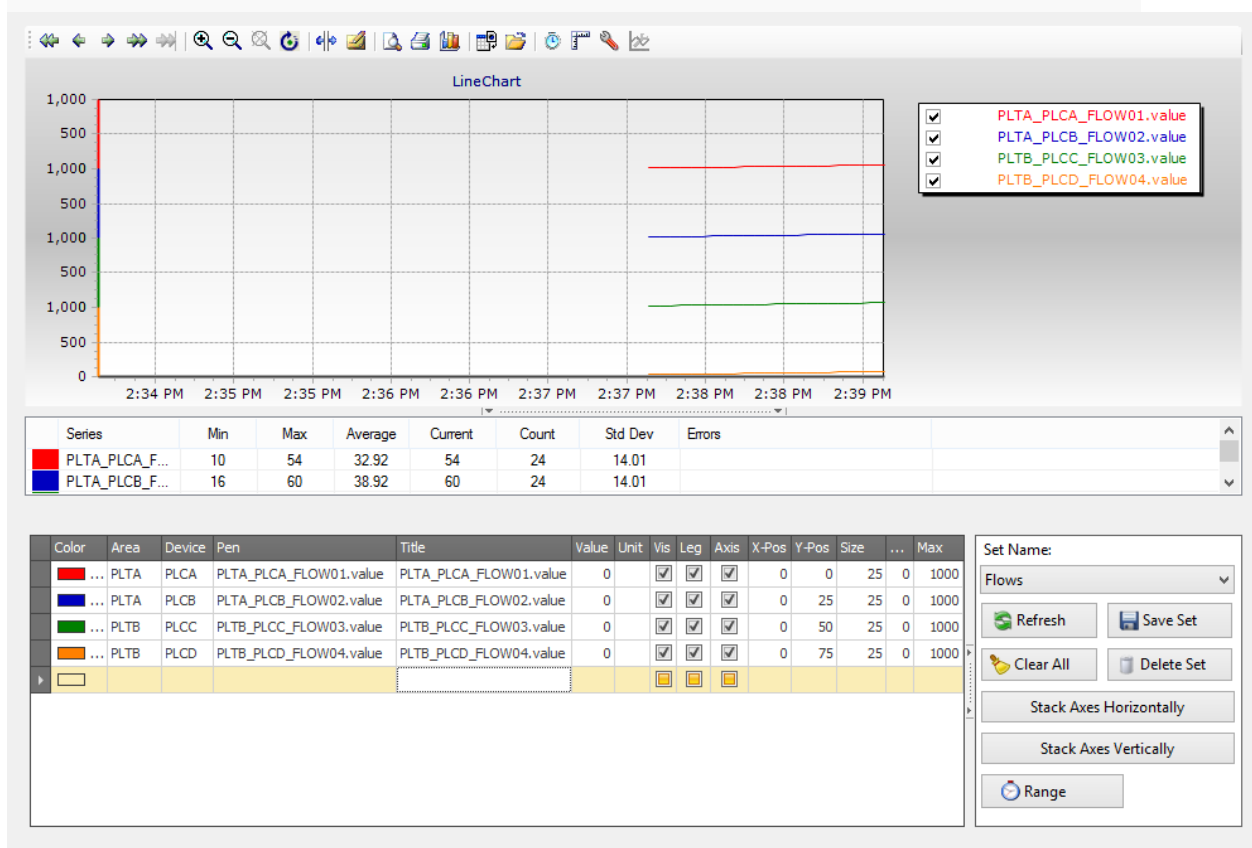
trendConfigurator.CurrentDevice	
Inputs:	
CurrentDevice	
Format	{0}
Outputs:	
CurrentDevice	
Event Trigger	
Event Object	trendConfigurator
Event Name	DeviceChanged

On the right, the **Objects** tree is visible, showing the **Misc** property group expanded to include **PenList**. Red arrows indicate the configuration steps: one arrow points from the **Return Object** output of the **TagsPerDevice** spider to the **PenList** property in the **Objects** tree, and another arrow points from the **Return Object** output to the **CurrentArea** output of the **trendConfigurator.CurrentArea** spider.

[Back to top](#)

Operate the TrendConfigurator control at runtime

Typical runtime configuration:



Note: By default, the run-time configuration of all TrendConfigurator controls are stored in a user profile variable called **TrendConfigurator**. Therefore if you want to maintain separate lists of sets for your TrendConfigurator controls, then rename the **UserProfileSetting** property (variable name) of each TrendConfigurator control.

For each **Pen** (tag or value) that you select to be trended as part of a set, you can specify the following:

IMPORTANT: You need to click the **Save Set** button (on the right of the TrendConfigurator control) after adding or configuring one or more series to this list, to save these changes to the current set (as specified by the **Set Name** combo box on the right of the TrendConfigurator control). Otherwise when you close this graphic form, the new series and/or configuration changes will be lost.

Note: You may need to click the **Refresh** button to view with any series that you add to the current set.

- **Color:** Use the color box to choose the required color for this series or value (and its axis color if it has its own custom axis) in the associated LineChart.
- **Area:** The specific plant area, in which this tag is physically situated.
- **Device:** The specific FED device in which this tag is being communicated.
- **Pen:** The name of the tag of this series.
- **Title:** Edit the name of the series or value that is displayed in the legend of the associated LineChart (if you choose this value to be visible in the legend by checking its **Leg** checkbox).

Note: The **UsePenDescForSeriesName** property determines whether the description of the selected tag or its tag name is used for the series **Title**. [Click for more details:](#)

If this property is True (the default selection) then the tag description is used, which is what the **Pen Description** combo box displays; otherwise if False then the tag name is used, which adheres to the following naming convention: AdroitAgentName.AdroitAgentProperty.

- **Value:** The current value of the tag being used for this series when it was selected.
- **Unit:** If necessary, specify the required unit to specify on the custom axis of the series on the LineChart for the trended value or series. If you are using MAPS, then this is retrieved from the MAPS database, for you.

- **Vis:** This checkbox toggles the series or value visibility on the associated LineChart. In other words it is displayed in the LineChart when checked and it is hidden when unchecked - the default selection.
- **Leg:** This checkbox toggles the series or value visibility in the legend of the associated LineChart. In other words it is displayed in the legend when checked and it is hidden when unchecked - the default selection.
- **Axes:** This checkbox determines whether this series or value will be charted using the default Y (or value) axis (when unchecked - the default selection) OR whether a custom Y axis will be used, when this is checked. If you choose to use a custom Y axis then use the remaining fields to configure the display of this axis in the associated LineChart, [as follows](#):
 - **X-Pos:** Specify the horizontal position of this custom Y axis, relative to the position of the default Y axis, as a percentage of the total horizontal width (size) of the default X axis. In other words, a value of a 100 moves the custom Y axis to the other side of the LineChart.

Note: A positive value shifts this custom Y axis to the right of the default Y axis, by the specified amount; while a negative value shifts this custom Y axis to the left of the default Y axis, by the specified amount.
 - **Y-Pos:** Specify the vertical position or topmost end of this custom Y axis, relative to the top of the default Y axis, as a percentage of the total vertical height (size) of the default Y axis. In other words, a value of a 100 hides this custom Y axis and therefore this series too.

Note: A positive value decreases the height of this custom Y axis below that of the default Y axis, by the specified height; while a negative value increases the height of this custom Y axis above that of the default Y axis, by the specified height.
 - **Size:** Specify the size of this custom Y axis, from the specified Y-Pos location in the direction of the X axis (lower-left corner), as a percentage of the total vertical height (size) of the default Y axis. In other words, a value of 100 is the size of the actual Y axis.
 - **Min:** The minimum value of this Custom Y axis, which depends upon the value being charted. By default this is set to 0, although, if possible, this control will obtain this value from the applicable Adroit datasource.

Note: Any series value below the specified **Min** value will be hidden from view.
 - **Max:** The maximum value of this Custom Y axis, which depends upon the value being charted. By default this is set to 100, although, if possible, this control will obtain this value from the applicable Adroit datasource.

Note: Any series value above the specified **Max** value will be hidden from view.

Using buttons, on the right of the TrendConfigurator control, you can change how the values of the current set are trended in the associated LineChart, as follows:

To view added series for the current set that are not yet displayed: click the **Refresh** button.

To set the time interval of the X-axis for all series:

Click the **Range** button, on the lower right of the TrendConfigurator control.

This displays the **Time Details Settings** dialog so you can configure the starting and ending times that apply to all the values that you are trending.

Note: This affects the time interval for all series of all the sets that you display on this LineChart until you close the Smart UI Operator, at which time this configuration will be lost.

The following axes-stacking actions change both the Custom and default Y axes parameters, which you can edit if you want to change their resultant settings:

Note: These Axes stacking actions only apply if you have configured one or more custom Y axis for this set of trended values. In other words, you have at least one **Axes** checkbox checked.

To view each custom Y axis on its own axis: This is particularly useful when you are trending values, whose custom Y axes have differing scales (differing maximum values), since this separates each custom Y axis for easier readability.

Click the **Stack Axes Horizontally** button, on the lower right of the TrendConfigurator control.

Note: This automatically configures the horizontal positioning (**X-Pos**) of the Custom Y axes for the best fit.

To trend each series using their own Y (value) axis: This allows you to compare the trend of each series (value) without confusing or hiding the trends of the other series.

Click the **Stack Axes Vertically** button, on the lower right of the TrendConfigurator control.

Note: This automatically configures the vertical positioning (**Y-Pos**) and **Size** of the Custom and default Y axes of each trended series, so that each series has their own Y axis that is the same size, one above the other.

Using buttons, on the right of the TrendConfigurator control, you can manage and configure sets of your values that are trended in the associated LineChart, as follows:

To create a new set:

1. Open the **Set Name** combo box and select the **New Set** entry.
2. Add and configure the series (values) that you want to trend in this set in the list on the left of the TrendConfigurator control.
3. Click the **Save Set** button to name this set and add it to the **Set Name** combo box.

To save and name a new set:

Click the **Save Set** button, which prompts you for the required name for this set of trended values, which is then added to the **Set Name** combo box.

Tip: Ensure that you provide a descriptive or unique enough name so that you will remember exactly which trend values this set contains.

To configure and/or trend the series (values) that you have added to a set:

Open the **Set Name** combo box and select the name of the required set.

This loads the series (values) and their configured properties in the list on the left of the Trend-Configurator control and then updates the associated LineChart to trend these series.

To update the current set with newly added and recently configured series:

Click the **Save Set** button.

This updates the set that is currently displayed by the **Set Name** combo box, with all the newly added series (values) and any changes that have been made to the properties of the other series in this list.

WARNING! If you forget to do this and then close this graphic form or exit the Smart UI Operator, then ALL the newly added series and/or configuration changes will be lost.

To programmatically select a specific set from the Set Name combo box:

Specify the name of the required set to load in the **TriggerLoadSetName** property and then trigger the Boolean **TriggerLoadSet** property via a spider, for instance by using a **Constant** spider, whose output is set to True.

This loads the series (values) and their configured properties in the list on the left of the Trend-Configurator control and then updates the associated LineChart to trend these series.

To remove all the series or values from the left list: click the **Clear All** button.

Note: This action does not automatically stop these values from being trended by the associated Line-Chart.

WARNING! After clearing all the series or values, ONLY click the **Refresh** button if you want to remove all the series or values from the set that is currently displayed by the **Set Name** combo box.

To delete the current set:

Click the **Delete Set** button.

This removes the set that is currently displayed by the **Set Name** combo box, from this list of sets, which therefore removes all of its added series (values) and their configuration.

Related information

Adding OLE DB datasources

Add an OLE DB datasource to expose the data contained within an OLE DB compliant database or data store. [Click for more information:](#)

Note1: This can only be performed if the currently logged on user has the necessary security permissions.

Note2: The required OLE DB data or service provider must be installed on both the Smart UI Client and Server computer, as BOTH computers need to access the required data store.



Note3: It is necessary to know all the information necessary to connect the required OLE DB data provider.

Note4: Once an OLE DB datasource has been added, each table is monitored by the Smart UI Server, at a configurable rate for new or updated data.

Tip: When editing an OLEDB datasource connection, the **Data Link Properties** dialog uses the previously supplied details, so you only need to specify the changed details NOT all of them.

To add and configure an OLE DB datasource

[Click for generic instructions on how to add and name a datasource:](#)

1. Open the **Enterprise Manager**.
2. If necessary, expand  **Servers**.
3. Right-click on the name of the required Smart UI Server or its icon  and select **Add Datasource**.

The **Add Datasource** configuration dialog is launched displaying the available datasource plugins.

Note: Only the datasource plugins, which have been licensed, appear in this list.

4. Click the datasource you require and then click the **Finish** button.

Tip: You can also simply double-click the required datasource.

The **Add A New Datasource** configuration dialog is displayed to configure this datasource:

5. Type an appropriate name for this datasource into the **Name** edit box.

When naming this datasource, [please be aware of the following:](#)

WARNING! Once a datasource has been named, it is NOT possible to rename it later.


Note1: The following non-standard text characters are NOT supported for datasource names: \ / : * ? < > | # "

Note2: Datasource names are case sensitive.

Note3: The specified name cannot be the same as any other datasource that has been already been connected to this Server.

Tip: Although the datasource name can be up to 50 characters long - use a descriptive name that is as concise as possible. This will ensure that users will BOTH easily identify and navigate to it within the **Enterprise Manager**.

To configure the OLE DB datasource

6. Connect to the required OLE DB data, by clicking the browse button  to the right of the **Connection String** edit box.

This will display the **Data Link Properties** dialog.

- a. Use the **Provider** tab to select the appropriate OLE DB provider for the type of data you want to access, from the list.

Note: This list contains all the OLE DB data or service providers currently installed on this Client computer, however, it is important to ensure that selected OLE DB data provider is ALSO installed on the required Server computer too.

- b. Click the **Next** button to display the **Connection** tab for the selected OLE DB provider. This tab is used to provide the necessary details to connect to the required data store. For more details,

click the **Help** button to view the provided help. Click **OK** when finished.

WARNING! Please ensure that the required security credentials (authentication) are always supplied when connecting to a remote computer, even if connection is correctly tested without specifying them. Since once the Server has successfully connected to a computer, any further connections made to this computer will work whether the correct authentication is supplied or not.


Note: If the required data store is password-protected, ensure that the **Allow Saving Password** checkbox is checked.

Tip: You can navigate directly to the **Connection** tab by double-clicking the desired provider.


7. Click the **Test Server Connection** button to ensure that the Server is able to successfully connect to this OLE DB data store.

If the connection was not tested successfully:

- a. Ensure that the connection to this data store has been correctly configured, by clicking the browse button to the right of the **Connection String** edit box and ensure that the **Connection** tab details have been correctly supplied.
 - b. Ensure that the Smart UI Server computer is also able to access this specified data store, since the connection is initially made relative to the Client computer.
8. Click **Finish** once the connection has been successfully tested and the datasource has been named.

This will add this specified OLE DB datasource to the **Datasources** category in the **Enterprise Manager**, which can be identified by its specified name or icon .

To display the connection details of an OLEDB datasource:

Right click the OLEDB  datasource in the Enterprise Manager and select the **Connection Details** option.

Note: If the connection string for this OLEDB datasource contains a password, it is hidden in *'s for security reasons.

Adding Adroit datasources

Add an Adroit datasource to connect to either a standalone Adroit Agent Server or a Adroit Cluster, to expose the configured agents and their slots. [Click for more information:](#)



Note1: Only users that have the necessary security permissions can do this.

Note2: You can only connect to an Adroit Agent Server on the Smart UI Server computer or a computer within its network domain.

Note3: When adding an Adroit datasource, one remote connection will be used, from the Adroit license of the connected Agent Server.

To add and configure an Adroit datasource

[Click for generic instructions on how to add and name a datasource:](#)

1. Open the **Enterprise Manager**.
2. If necessary, expand  **Servers**.
3. Right-click on the name of the required Smart UI Server or its icon  and select **Add Datasource**.
The **Add Datasource** configuration dialog is launched displaying the available datasource plugins.
Note: Only the datasource plugins, which have been licensed, appear in this list.
4. Click the datasource you require and then click the **Finish** button.

Tip: You can also simply double-click the required datasource.

The **Add A New Datasource** configuration dialog is displayed to configure this datasource:

5. Type an appropriate name for this datasource into the **Name** edit box.

When naming this datasource, [please be aware of the following:](#)

WARNING! Once a datasource has been named, it is NOT possible to rename it later.


Note1: The following non-standard text characters are NOT supported for datasource names: \ / : * ? < > | # "

Note2: Datasource names are case sensitive.

Note3: The specified name cannot be the same as any other datasource that has been already been connected to this Server.

Tip: Although the datasource name can be up to 50 characters long - use a descriptive name that is as concise as possible. This will ensure that users will BOTH easily identify and navigate to it within the **Enterprise Manager**.


To configure an Adroit datasource

6. Select the Agent Server type, as follows:
 - a. **Standalone:** This should be used when connecting to a single standalone Agent Server that is running the required agent database containing the configured agents and slots.
This requires specifying this Agent Server in the **Agent Server** drop down selection box.
WARNING! This must be specified and cannot be left blank.
 - b. **Redundant:** This should be used when connecting to an Adroit Cluster i.e. a set of two Agent Servers each running a replicated agent database.
This requires specifying the names of both cluster partners in the **Agent Server** and **Partner Server** drop down selection boxes provided.
To use a drop down selection box to specify an Agent Server:
Click the  button to the right of the required drop down selection box.
This process depends upon whether the required Agent Server is displayed in the list box or not, as follows:


Note: The list box is populated by all the Adroit Agent Servers on the Server computer and within its network domain.

 - i. If the required Adroit Agent Server is displayed in the list box, select it to immediately add it to the drop down selection box.
Note: If the required Agent Server has been started after opening this dialog, click the button to the right of the drop down selection box to update this list.
 - ii. If the required Adroit Agent Server is not specified in the list box, then type it into the drop down selection box, using the following format: **ComputerName.AgentServerName**. Where:
ComputerName is the network or computer name of this Agent Server, if required.
AgentServerName is the name used to uniquely identify this Agent Server. This is the first name displayed in the title bar of the Adroit Agent Server window.
Note: The onus is upon the user to ensure that the specified computer name and Agent Server name is correct.
7. Click **Finish** once the required Agent Server has been specified and the datasource has been named.

This adds an Adroit datasource to the **Datasources** category in the **Enterprise Manager**, identified by either the specified name or icon .

Note: If the added Adroit datasource is identified by this icon , there is a problem connecting to the specified Agent Server. For details, see [Troubleshooting Adroit datasources](#).


Troubleshooting Adroit datasources


If after adding an Adroit datasource, it is identified by this icon  in the **Enterprise Manager**, then there is a problem connecting to the specified Agent Server.

The following list indicates the possible causes for the Server not being able to connect to the specified Agent Server and their remedies:

Cause	Remedy
Agent Server has been shut down	Determine status of Agent Server and restart if required.
Network connection has been lost	Determine network status, and reconnect if required.
Agent Server has an exception (an error)	Restart the Server.

Select queries

Queries provide common tasks for managing your database that is connected to an  OLE DB datasource.

The  **Select Queries** can select a sub-set of your data by tailoring your data to suit specific uses or requirements. [Click for more information:](#)

In addition to selecting the required tables and/or views and their fields, you can also create relationships between fields in these tables and/or views.

You can also add criteria that must be met by values in one or more fields. If necessary, parameterize the criteria values to specify them before performing the query.

Note1: Unlike tables and databinders, the data obtained by a Select query is not updated by the Server and remains as current as the last time that it was executed.





Note2: Unlike tables that ONLY display their last 1000 records, Select queries display ALL the records that match its specified criteria.


Note3: Select queries can be queued to ensure that a query will not be lost when the connection to its database fails.

To add and configure a Select query:

Add a Select query:

[Click for generic instructions on how to add, name and specify the creation method for a query:](#)

1. Open the **Enterprise Manager**.
 2. If necessary, expand  **Servers**.
 3. If necessary, expand the name of the required Smart UI Server connection .
- Tip:** This connection name is prefaced by the Smart UI Server computer name.
4. If necessary, expand the **Datasources** category.
 5. If necessary, expand the required OLE DB datasource name or its icon .
 6. Right-click **Queries** or its icon  and select the **Add XXX Query** option, where **XXX** is the required query category.

Tip: Alternatively, simply double click the subfolder  of the required query category .

7. [Specify the required Query Name in the Add XXX Query configuration dialog:](#)

WARNING! Once a query has been named, it is NOT possible to rename it later.

Note: The specified name cannot be the same as any other query that has been already been specified.>

Tip: Give this query a meaningful and concise name so that its purpose can be clearly identified - try not to use all 55 characters.

8. If supported by this query category - choose how you want to create this query, [by selecting one of the following options:](#)

- **XXX Query Wizard:** This default option uses a structured configuration dialog to create the query. This method is especially recommended for those who are not familiar with the SQL querying language.

Tip: Advanced users can use the configuration dialog to perform the basic configuration such as specifying the required tables and their fields and then customize the added commands using the final dialog, which is the SAME dialog provided by the **Custom Query** option.

Note: This generates a generic SQL statement and will work with all types of OLE DB databases.

- **Custom Query:** This method is intended for those users who are already conversant with the SQL querying language and want to quickly create their query. For details, see [Creating customized queries](#).

9. Click **Next**, once the query is named and the required method of its creation is selected.

Configure this Select query:

10. **Select the required fields** from the **Tables and Fields** tree in the **New User Defined Query** dialog.

Check the checkbox of each table and their field/s that you would like to be queried, by clicking on it.

If necessary, expand the **Views** node and check the checkbox of each view and their field/s that you would like to be queried, by clicking on it.

The names of tables and views support SQL schemas. Click for more details: The default schema 'DBO' is NOT displayed. But if a table or view is part of a different schema, then the name of this schema is indicated in brackets after the name of the table or view. This means that a table or view can have the SAME name as another table or view as long as it belongs to a different schema.

Tip: Double-click on the name of a table or view to collapse it.

As you select the fields, the appropriate SQL query commands are added to the **SQL Query** field, which cannot be edited.

Tip: Click the **Preview** button to display the fields that will be returned by this query, in the pane at the bottom of the configuration dialog.

WARNING! If no fields / tables / views are selected, the query cannot be created and the **Preview** button will display, the following **Error: "Syntax error in query. Incomplete query clause."**

Ensure that you select BOTH the fields that you wish the query to display and those that are required to refine this data (such as those you want to use to create relationships between and define constraints for).

Note: This is the only dialog that must be completed, within the query configuration dialog, so you can click the **Finish** button at this point.

11. Click **Next**, once the required fields are selected.


12. If necessary, add or remove relationships between the selected tables or views in the **Relationship Setup** dialog.

Typically, you create relationships between tables in order to perform lookups on fields between different tables such as between the foreign key of one table and the primary key of a related table.

Adding or removing relationships:

In this case a relationship is simply a pair of selected fields, usually in different tables or views, whose values are required to be identical in order for a record to be returned.

To add a relationship:

- Click the **Add Relationship** button to create a relationship between two of the selected fields, as follows:
- Click the drop-down list box button  at the right of each **Field** list and select the required field from both tables or views. This list only displays the fields selected in the previous dialog, in the following syntax: [Tablename].[Fieldname] or [View-name].[Fieldname].

Note: If necessary, you can edit the selected table, view and/or field names yourself - in this case ensure that you spell these names correctly.

To delete the last relationship:

Click the **Delete Relationship** button to remove the last relationship from the Relationships list.

WARNING! This will ALWAYS delete the relationship at the bottom of the list, regardless of the currently selected item.

Note: Ensure that all the relationships are fully defined, otherwise the query will also not work i.e. delete any partially configured relationships.

Tip: Press the **Preview** button to test your defined relationships, since an exception will be displayed if any of the data types of the related fields are NOT identical.

As you define relationship(s), the appropriate SQL query commands are added to the SQL Query field, which cannot be edited.

Note: These do not affect the relationships that you may have already created within your database.

13. Click **Next**, once the required relationships have been defined.
14. If necessary, add or remove constraints for the values of one or more of the selected fields in the **Constraints** dialog.

Typically, you constrain values of one or more fields to prevent this table from being needlessly updated. If these constraints will change from user to user then use parameters instead of hard-coded values.

Adding or removing constraints for the fields of this table:

To add a constraint:

- a. Press the **Add Constraint** button.
- b. If this constraint is NOT the first, specify how to relate this constraint to the previous constraints, as follows:
 - **AND:** This is the default option, which makes this constraint mandatory. For instance, if there are two constraints with an AND between them, then the data will be restricted by BOTH constraints.
 - **OR:** Use this option, to make the specified constraint optional. For instance, if there are two constraints with an OR between them then the data can be restricted with EITHER of these constraints.
- c. Select the field, whose values you want to limit, from the **Field** drop-down list box, in the following syntax: [Fieldname].
- d. Select the method by which the values of this field are to be constrained or limited from the **Constraint** drop-down list box. For details, see [Available constraint operators](#).
- e. Specify the constraining value or parameter in the **Value** field.

Note1: This list only displays the selected fields.

Note2: If necessary, you can edit the selected field name yourself - in this case ensure that you spell these names correctly.

Specify a value if your users will not need to change this criterion, otherwise specify a parameter if your users will need to change this criterion, for instance, if different users require different values. For details, see [Adding parameters](#).

Note: Ensure that the specified **Value** has the same data type as the specified **Field**, otherwise the query will not work.

To delete the last constraint:

- a. Click the **Delete Constraint** button to remove the last constraint from the Constraints list.

WARNING! This will ALWAYS delete the constraint at the bottom of the list, regardless of the currently selected item.

Note: Ensure that all the constraints are fully defined, otherwise the query will also not work i.e. delete any partially configured constraints.

As you define the constraint(s), the appropriate SQL query commands are added to the SQL Query field, which cannot be edited.

WARNING! If you need to configure or modify any of the previous pages, click the **Previous** button to navigate to these pages. Once you click the **Next** button the query configuration dialog ends and any further modifications to the query can only be performed manually.

15. Click **Next**, once the required constraints have been defined.

16. If necessary, define the sorting order of the resultant records in the **Ordering** dialog.

Typically, you sort a query to ensure that the records are displayed in order of importance or relevance.

Although more than one field can be used to sort the data, the closer the field is to the top of the list, the greater the impact it has upon the sorting process.

Typically multiple fields are used to sort queries, when their entries are related – for instance can be used when sorting by surnames and then sorting by their first names. It may also be useful to sort by identity (ID) fields or by Country, when viewing a list of orders etc.

Adding or removing sorting orders:

To add a sorting order:

- a. Press the **Add Order** button.
- b. Select the field, whose values you want to sort, from the Field drop-down list box.

Note1: This list only displays the selected fields.

Note2: If the View is selected the syntax for the fields is [tablename].[fieldname] or [Viewname].[fieldname].

- d. Select the method by which the values of this field are to be sorted from the Constraint drop-down list box, as follows:
 - o ASC: The default value, which sorts the values of the specified field in ascending order. The method used to sort the values, depends on the data type of the specified field, as follows:
 - o **String:** This displays records beginning with non-text characters such as `'_` or `!`, then records from 0 to 9, then records from A to Z, with lowercase letters preceding uppercase letters e.g. for text like fgfd and Fgfd, the ASC order is fgfd and then Fgfd.
 - o **Numeric:** This displays records from 0 to 9.
 - o **Date Time:** This will display the records in this field from newest to oldest.
 - o DESC: This sorts the values of the specified field in descending order, which is the reverse of the ASC order.

To delete a sorting order:

- a. Specify the required View or the name of the Table in the Select View/Table list box.
- b. Click the Delete Sort Order button to remove the last sort order from the Sort Orders list.

WARNING! This will ALWAYS delete the sort order at the bottom of the list, regardless of the currently selected item.

As you define the sort order(s), the appropriate SQL query commands are added to the SQL Query field, which cannot be edited.




17. Click **Next**, once the required sort orders have been set up, to end the query configuration dialog.

The **Custom Query** dialog is displayed, which allows you to manually configure this query, if necessary.

18. If necessary, type or modify the existing SQL querying commands in the **SQL Query** editor, which highlights keywords and makes it easier to customize your queries.

Note: The **Preview** button is as easy way to debug your query as you manually edit it. Clicking this button will return all the fields, in the pane at the bottom of the configuration dialog, that the current query would return. In this way you can ensure that your query does return what you want it to.

19. Click **Finish**, once you are satisfied with this query.

This adds this query with its specified name beneath the  **Select Queries** subfolder of the  **Queries** folder of the applicable  OLE DB datasource, in the **Enterprise Manager**.



If using SQL, please ensure that all time based queries adhere to the universal date time standard for SQL, which is yyyy-mm-dd, as follows:

- Either, set the short date format in Regional Settings to be yyyy-mm-dd, once for all.
- Or convert ALL the date parameters for each query into this SQL standard format first, by using a string converter, such as the String Format spider.

To perform a Select query using a Query spider: this procedure describes how to perform a Select query by clicking a Button control, so its results can be displayed on a graphic form within a DataGrid control.

This assumes that the Select query has been added and configured.

Prepare the graphic form:

1. Open a new graphic form in the **Design workspace**. (If necessary, add a project and a graphic form first).
2. Add a  Button control to this graphic form from the **Toolbox** - this will be used to trigger the Query spider so that it will perform the query.
3. Add a  DataGrid control to this graphic form from the **Toolbox** - this will be used to display the records returned by the Select query.
4. If necessary, select the DataGrid control in the graphic form.

Add and configure the Query spider to run this query:

5. Open the **Spider Configuration** tool window.
6. Right-click the spider workspace itself and select the **Create New Spider** menu.
7. Click the **Databasing** sub-menu and its **Query** item to add a Query spider.
8. Double click the title bar of this spider to open its configuration view.
9. Select the required query from the **Inputs** section of this spider, as follows:
 - a. Click the Server containing the required OLE DB datasource from the **Server** list box.
Tip: Use the **Default Server** selection, when referring to the Server that you use to log in, instead of its actual name as this makes it easier to transfer this spider to another Client.
 - b. If necessary, select the required **OLE DB datasource** from this list of available OLE DB data-sources on this Server.
 The first OLE DB datasource created on this Server is selected by default.
 - c. Click the required type in the **Query Type** list.
 The Select Query category is selected by default.
 - d. From the **Query Name** list box, select the name of the required query.
Note: It is important to name your queries clearly, so that it is easy to select the correct one.
 - e. If this selected query contains parameters, then the names of these parameters will be added as additional inputs to the **Inputs** section.
Note1: The value of each parameter input can either be provided by a property or data element via a silk or can be typed in manually.
Note2: If using SQL, please ensure that all the date parameters for this query adhere to the universal date time standard for SQL, which is yyyy-mm-dd.
Tip: When typing in a value manually, double-click the value column to the right of the parameter name in the **Inputs** section and type in the required value.
 - f. It is recommended to use the **Confirm Action** checkbox to display a confirmation dialog before this query inserts a record into your database.

This spider performs this query to insert a record, when it is triggered.

WARNING! Please ensure that you have provided values for ANY parameters used by this query BEFORE triggering this spider otherwise an error will be displayed.

Note: You will notice in the **Event Trigger** section of this spider that this Query spider is triggered by clicking the **Button** control.

Connect the Query spider to the DataGrid control:

10. If necessary, open the property tree by using the pane display bar on the right border of the central spider workspace pane.
11. In the **Outputs** section, drag the **Return Object** output onto the **Data** category beneath this Data-Grid control in the property tree and when this category expands, drop it onto the **DataSource** property.

Note: Ensure that the **Success** output is True (i.e. the query ran successfully), if this is False then there is either a problem with the query syntax or another problem preventing this query from running.


Tip: The **Rows Exist** output is only True when the query returns data and is False if an empty dataset is returned.

Additional filtering and messaging of the displayed data:


12. This launches the **Filter DataSet** configuration dialog, whose first dialog can be used to select which of the selected tables and their fields you want to display. By default all of the tables and fields are selected.

Note: If this configuration dialog does not launch automatically, double-click the **DataSource** property in the property tree and right-click this spider's title bar and select **Filter DataSet** from this menu.
13. If necessary, click **Next** to display the **Filter** dialog, which typically should not be required since the Select query allowed you to define your own constraints. Therefore, click **Next** to display the **Sort By** dialog, in which you click the column heading that you want to sort the columns by in ascending or descending order and then click **Finish**.

Test that the query is correctly displayed:

14. Click the **Preview** toolbar button of this graphic form.
15. Click the Button control and see if the DataGrid control correctly displays the results of this query.
16. Save the graphic form by pressing its **Save**  toolbar button to save these changes.

Creating customized queries

When adding a query to an OLE DB datasource , you can either use the structured configuration dialog or, if you are familiar with the SQL querying language, simply create your own custom query directly.

To create a custom query:

This procedure assumes that the required query has already been added and the query configuration dialog is launched.

1. Specify the required **Query Name**. [Click for more details](#):

WARNING! Once a query has been named, it is NOT possible to rename it later.

Note: The specified name cannot be the same as any other query that has been already been specified.

Tip: Give this query a meaningful and concise name so that its purpose can be clearly identified. Try not to use all 55 characters.

2. Select the **Custom ... Query** option and perform any other configuration that is available in this dialog.
3. Click **Next**.

The **Custom Query** dialog is displayed, which allows you to manually create and configure this query.



4. Type in the required commands to perform the required task in the **SQL Query** editor, which highlights keywords and makes it easier to customize your queries.

When creating constraints within this query, parameters can be added so that their requirements can be specified before the query is performed. For details on specifying and using parameters, see [Adding parameters](#).

Note1: The **Preview** button, provides an easy way to debug your query as you manually edit it. Clicking this button will return all the fields, in the pane at the bottom of the configuration dialog, that the current query would return. In this way you can ensure that your query does return what you want it to.

Note2: When previewing queries containing parameters, a dialog box will be displayed for each parameter, allowing you to provide a temporary value for them.

- Click **Finish**, once you are satisfied with this query.

This adds this query to the applicable category of the  **Queries** folder beneath the  OLE DB datasource, in the **Enterprise Manager**.

Note: If using SQL, please ensure that all time based queries adhere to the universal date time standard for SQL, which is yyyy-mm-dd.

Available constraint operators

The **Constraints** dialog specifies requirements that the values of the selected fields must meet before they can be selected.

The following methods can define how the specified Value (requirement) relates to the value of each Field:

Operator	Name or Meaning	Usage
=	Equal to	If Field = value, then row is displayed.
>	Greater than	If Field > value, then row is displayed.
<	Less than	If Field < value, then row is displayed.
>=	Greater than or equal to	If Field >= value, then row is displayed.
<=	Less than or equal to	If Field <= value, then row is displayed.
<>	Not equal to	If Field <> value, then row is displayed.
!<	Not less than	If Field is not < value, then row is displayed.??
!=	Not equal to (the same as <>)	If Field is not equal to value, then row is displayed.
!>	Not greater than	If Field is not > value, then row is displayed.
EXISTS (sub-query)	The 'subquery' argument of EXISTS is a SELECT statement	The subquery is evaluated to determine whether it returns any rows. If the subquery returns at least one row, the result of EXISTS is TRUE.
LIKE	A character string comparison for SIMILAR pattern matching.	LIKE "string" searches for values matching specified "string". If Field = string, then row is displayed.
NOT LIKE	A character string comparison for EXCLUSIVE pattern matching.	NOT LIKE "string" searches for values not matching specified "string". If Field <> string, then row is displayed.

The "string" specified for the LIKE and UNLIKE comparison operators support the [following wildcards](#):

- '%' can be used as a placeholder for ANY number of characters. For instance "AB%" returns fields whose values are: "AB", "ABC", "ABCD" etc.
Tip: "%AB%" returns all fields containing "AB" anywhere in their values.
- '_' can be used as a placeholder for a SINGLE character'. For instance "AB_" returns fields whose values are: "AB", "ABC" but NOT "ABCD".
- '[]' specifies the required RANGE for a SINGLE character. For instance "[C-G]AB" OR "[CDEFG]AB" returns fields whose values ending with 'AB' and beginning with any single character between C and G, for example CAB, FAB etc.
- '[^]' specifies any SINGLE character NOT within the specified RANGE. For instance "[^C-G]AB" OR "[^CDEFG]AB" returns fields whose values ending with 'AB' and that do not start with any single character between C and G, for example BAB, PAB etc.

Note1: Wildcards can be inserted in ANY position within the string. For example, 'A%B'.

Note2: It is possible to combine these wildcards into wildcard expressions. For instance "de[^I]%" returns all fields beginning with 'de', where the following letter is not 'I'.

Operator	Name or Meaning	Usage
IS	IS	If Field = value, then row is displayed.
IS NOT	IS NOT	If Field <> value, then row is displayed.
IN	IN	IN expression , defines the expression in which all valid values are located. If Field is found in expression , then row is displayed.
NOT IN	NOT IN	IN expression , defines the expression in which all invalid values are located. If Field is not found in expression , then row is displayed.
BETWEEN	BETWEEN	BETWEEN begin_expression AND end_expression defines the range of valid values. If Field >= begin_expression and <= end_expression , then row is displayed.
NOT BETWEEN	NOT BETWEEN	NOT BETWEEN begin_expression AND end_expression defines the range of invalid values. If Field < begin_expression and > end_expression , then row is displayed.

Adding parameters

Use parameters when you or your users need to change the value of something frequently, such as a constraint. In other words, in every situation where a hard-coded value is too inflexible.

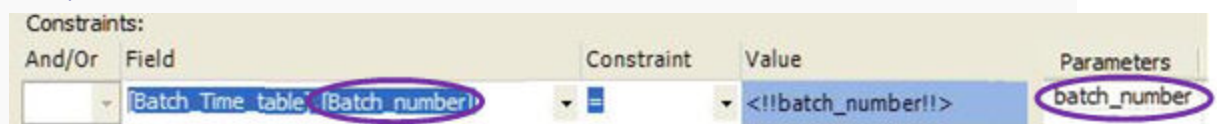
Parameters are essentially place-holders for values that can be dynamically specified, for instance when different users require different values.

To add a parameter:

- Right-click and select the required value placeholder as follows:
 - Add String Parameter:** A placeholders for textual values.
 - Add Generic Parameter:** A placeholder for values of other data types, such as numeric or date time values.
 - Add SQL DateTime Parameter** or **Add Access DateTime Parameter** A database-specific placeholders for date time values.

The default parameter name is the field name of the selected table.

Example:



Tip: Rename this default parameter to make it easier to recognize, when assigning a value to it, by manually editing the parameter name, taking care not to remove the encompassing '<!!' and '!!>' and any additional data-type formatting characters. You can also double click the name in the **Parameters** list.

- If necessary, specify the following data type-specific formatting for this added parameter:

Data Type	Additional Formatting	Example
Numeric	None	<!!parameterName!!>
String	Enclose the inserted parameter in single quotation marks (""") Note: This additional formatting can be added automatically by selecting Add String Parameter .	'<!!parameterName!!>'
DateTime	Enclose the inserted parameter in hashes ("#")	#<!!parameterName!!>#