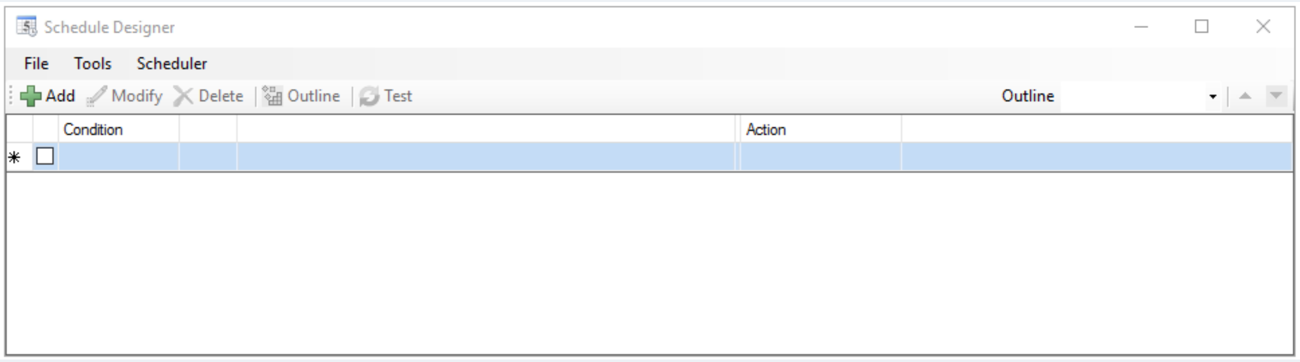
Event-Based Reporting

Overview

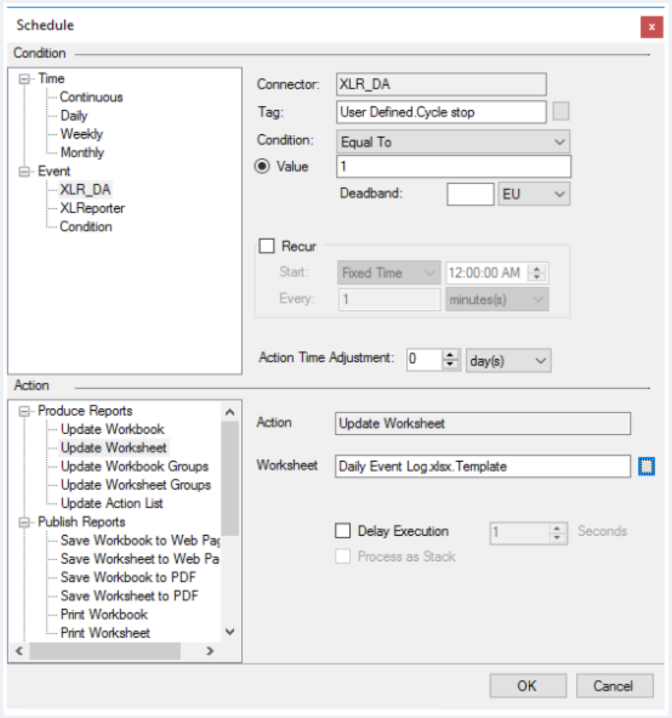
In many reporting applications, the most accurate way to collect data is based on a "trigger" condition in the process. In other words, the live KPI in a PLC may be most relevant at the exact moment in time when a machine has completed a job, or a trend graph must display data only for the duration of that job. Other times, there are multiple events occurring in a process that must be included, in time-series order, on a summary report. All these scenarios, and many more, can be handled in **XLReporter** by implementing an **Event-Based** schedule.

Schedule

At the highest level, event-based reports are driven from the **XLReporter Project Schedule**. The schedule organizes automated reports for the scope of the entire reporting application and can be configured from the **Project Explorer** under the **Project** tab by selecting **Schedule**, **Designer.**

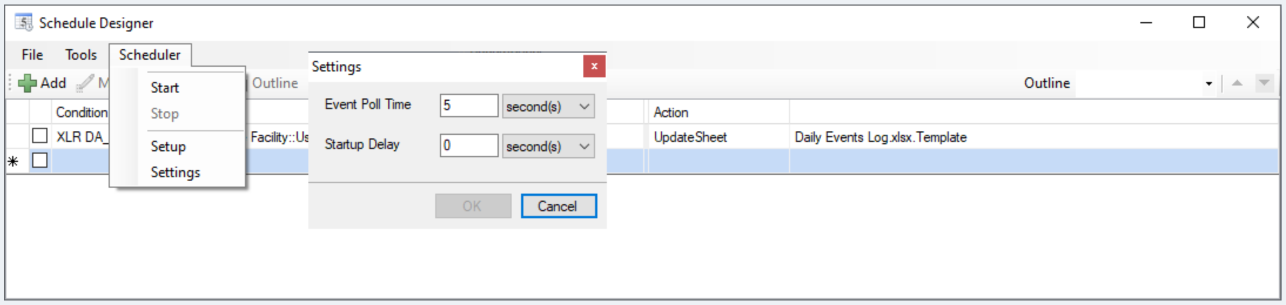


The **Add** button is used to add a new **Schedule Action.** Each **Action** has an associated **Condition**, which can be based on the system time, or based on a tag value read through a **Connector** defined for a real time source, like a PLC or HMI.



Connector: XLR_DA
Tag: User Defined.Cycle stop
Condition: Equal To
Value: 1
Deadband: EU
Recur
Start: Fixed Time 12:00:00 AM
Every: 1 minutes(s)
Action Time Adjustment: 0 day(s)
Action: Update Worksheet
Worksheet: Daily Event Log.xlsx.Template
Delay Execution 1 Seconds
Process as Stack

In the example image above, a **Daily Event Log** report is scheduled to update whenever the value of the **Cycle stop** tag transitions from any other value to the value of *1*.



Settings

Event Poll Time: 5 second(s)

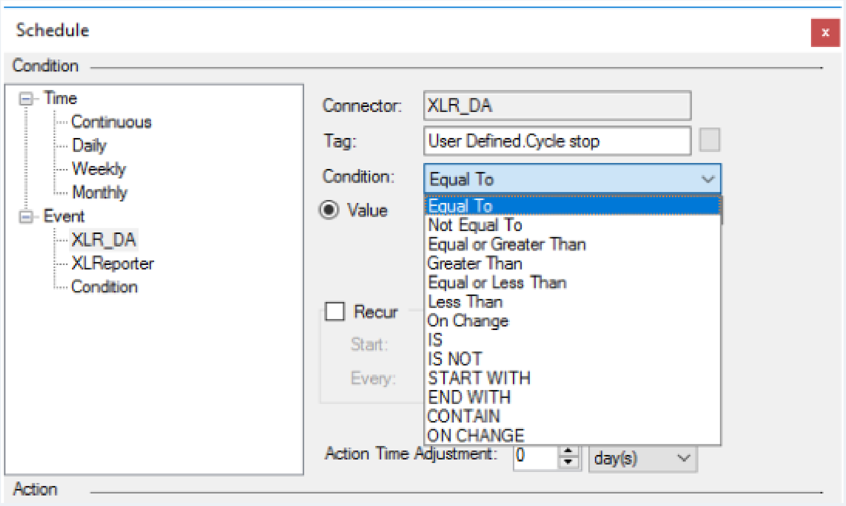
Startup Delay: 0 second(s)

Condition		Action	
XLR DA_	Facility::Us	UpdateSheet	Daily Events Log.xlsx.Template

With Event-Based scheduling, the **XLReporter Scheduler** polls the tag(s) set as conditions every **5 seconds** by default. This can be lowered to 1 second if necessary. If a transition into the **Condition** value from another value (such as a transition *from* 0 *to* 1) is detected from one polling cycle to the next, the **Action** is processed.

Event-Based Conditions

There are a variety of configurable **Event Conditions.**



Connector: XLR_DA

Tag: User Defined.Cycle stop

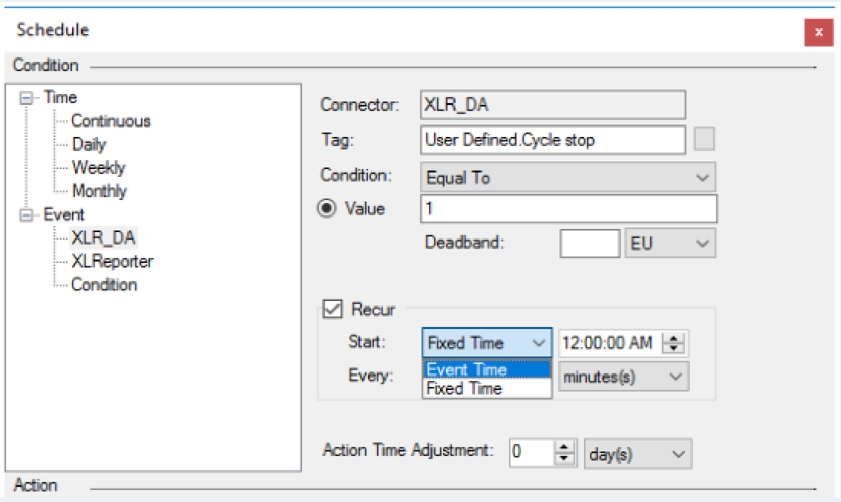
Condition: Equal To

Value

- Equal To
- Not Equal To
- Equal or Greater Than
- Greater Than
- Equal or Less Than
- Less Than
- On Change
- IS
- IS NOT
- START WITH
- END WITH
- CONTAIN
- ON CHANGE

Action Time Adjustment: 0 day(s)

The first half of conditions, which are spelled out with mixed casing, e.g., “Equal To”, operate on numeric tags such as floating point, integer, or Boolean values. The second half, spelled out in ALL CAPS, operate on string-based tags.



Connector: XLR_DA

Tag: User Defined.Cycle stop

Condition: Equal To

Value: 1

Deadband: EU

Recur ✓

Start: Fixed Time 12:00:00 AM

Every: Event Time / Fixed Time minutes(s)

Action Time Adjustment: 0 day(s)

The **Recur** parameter acts similarly to a *while* loop in that once the **Condition** is detected, as long as it remains true, the **Action** executes at the specified interval until the **Condition** is no longer true. The recurrence can be based on the time of the **Event Condition** or on the **Fixed** system time.

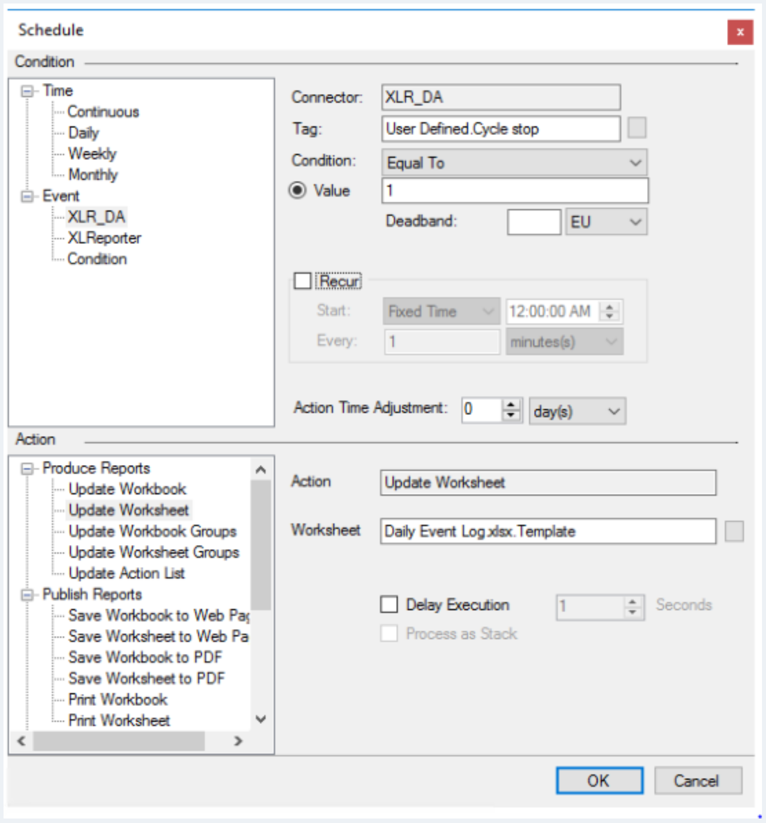
Event-Based Report Scenarios

Below are a few scenarios where event-based reporting is used.

Daily Cycle Report

In this scenario a report must be generated for the day displaying values from the process at the end of every cycle ran for the day.

Schedule



Schedule

Condition

- Time
  - Continuous
  - Daily
  - Weekly
  - Monthly
- Event
  - XLR_DA
  - XLReporter
  - Condition

Connector: XLR_DA

Tag: User Defined.Cycle stop

Condition: Equal To

● Value: 1

Deadband: EU

☐ Recur

Start: Fixed Time 12:00:00 AM

Every: 1 minutes(s)

Action Time Adjustment: 0 day(s)

Action

- Produce Reports
  - Update Workbook
  - Update Worksheet
  - Update Workbook Groups
  - Update Worksheet Groups
  - Update Action List
- Publish Reports
  - Save Workbook to Web Pag
  - Save Worksheet to Web Pa
  - Save Workbook to PDF
  - Save Worksheet to PDF
  - Print Workbook
  - Print Worksheet

Action: Update Worksheet

Worksheet: Daily Event Log.xlsx.Template

☐ Delay Execution 1 Seconds

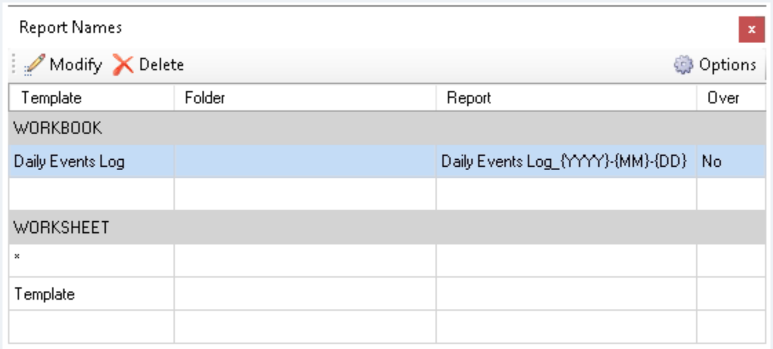
☐ Process as Stack

OK Cancel

The **Condition** used on the schedule represents the end of the process event being monitored, e.g., *Cycle stop =1*.

Naming Convention

Because the events are collated into separate worksheets based on the date, the **Workbook Name** uses the **Calendar Variables** *{YYYY}, {MM}, and {DD}*.



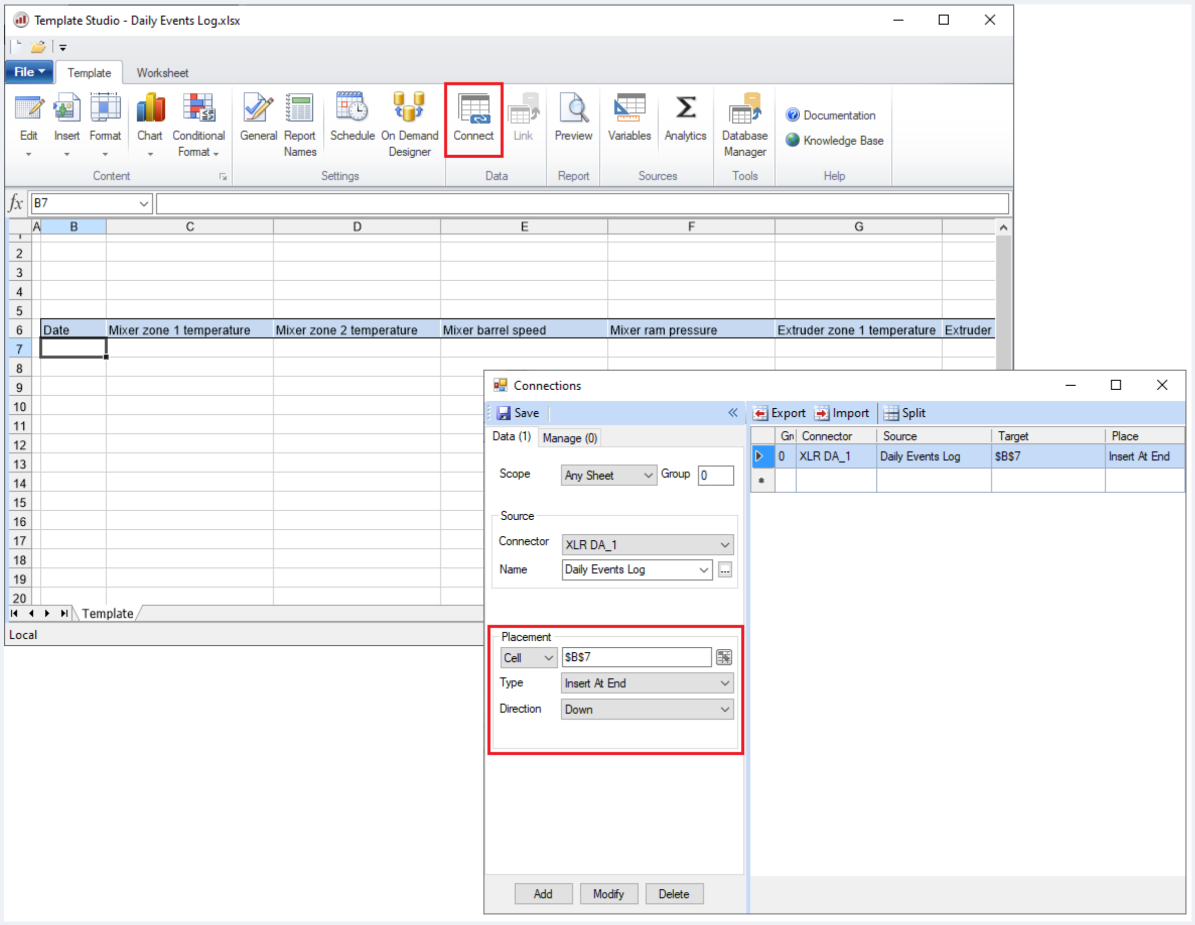
Report Names

Modify Delete Options

Template	Folder	Report	Over
WORKBOOK			
Daily Events Log		Daily Events Log_{YYYY}-{MM}-{DD}	No
WORKSHEET			
*			
Template			

Data Placement

When the report is updated, the real time data must be placed onto the next available row below the existing data. For that reason, the **Placement Type** used is *Insert at End, Down.*



B7

A	B	C	D	E	F	G	
6	Date	Mixer zone 1 temperature	Mixer zone 2 temperature	Mixer barrel speed	Mixer ram pressure	Extruder zone 1 temperature	Extruder

Connections

Data (1) | Manage (0)

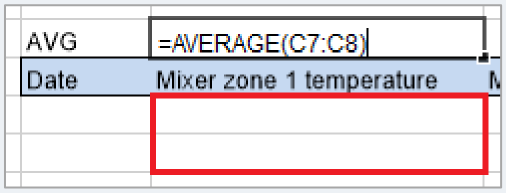
Scope: Any Sheet Group: 0

Source
Connector: XLR DA_1
Name: Daily Events Log

Placement
Cell: B7
Type: Insert At End
Direction: Down

Gr	Connector	Source	Target	Place
0	XLR DA_1	Daily Events Log	B7	Insert At End

Charts, Formulas and Formatting



AVG	=AVERAGE(C7:C8)
Date	Mixer zone 1 temperature

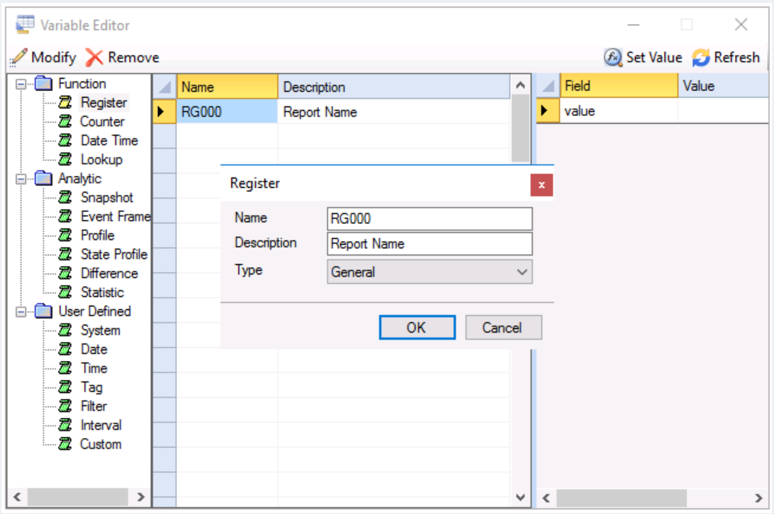
The data is inserted in *B7*. If any formulas, chart series, or formatting (including conditional formatting) are configured for rows 7 and 8, the formula or series ranges will automatically expand as each event row is collected and inserted into the report. In other words, configuring these functions to the first two data rows allows them to adapt dynamically to variable row counts. This is important because we do not know how many cycles will run each day.

Cycle Report

In this scenario a report is required for each cycle that captures the cycle ID at the beginning of the cycle and adds data to the report continuously throughout the cycle.

Variable

This configuration requires a **Variable** to be added to the project to manage the naming convention of the report. A **Variable** is added from the **Data** tab of **Project Explorer** by clicking **Variables** to open the **Variable Editor**.



Name	Description
RG000	Report Name

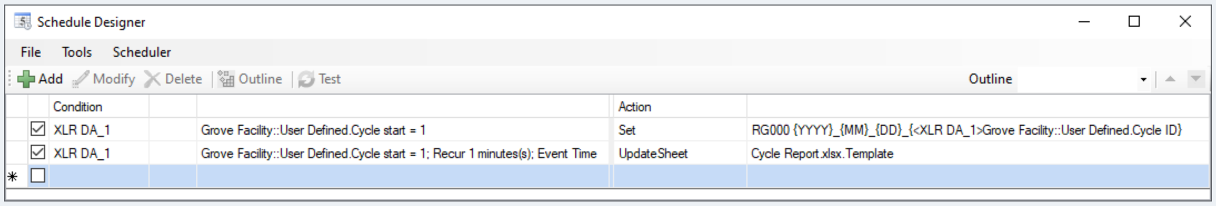
Field	Value
value	

Register

Name: RG000
Description: Report Name
Type: General

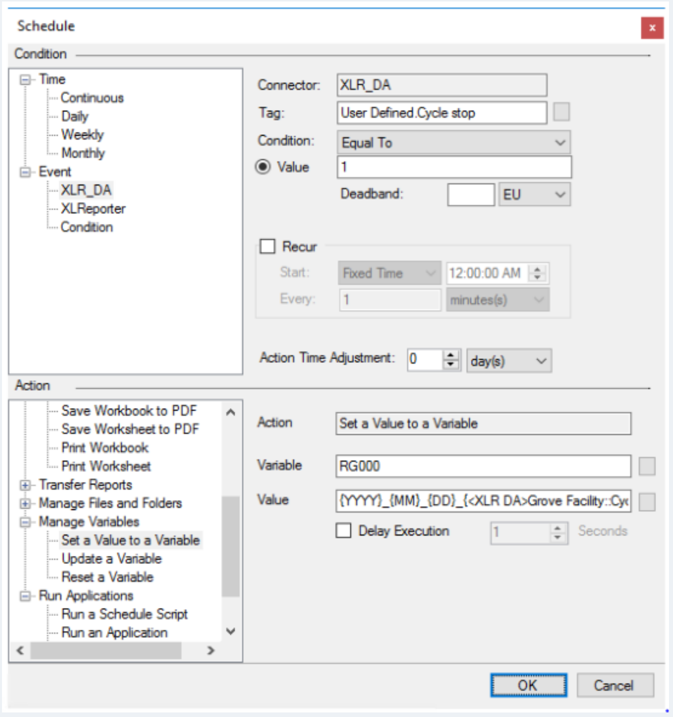
Register variables are provided to hold information like this.

Schedule



	Condition			Action	
☑	XLR DA_1		Grove Facility::User Defined.Cycle start = 1	Set	RG000 {YYYY}_{MM}_{DD}_{<XLR DA_1>Grove Facility::User Defined.Cycle ID}
☑	XLR DA_1		Grove Facility::User Defined.Cycle start = 1; Recur 1 minutes(s); Event Time	UpdateSheet	Cycle Report.xlsx.Template
☐					

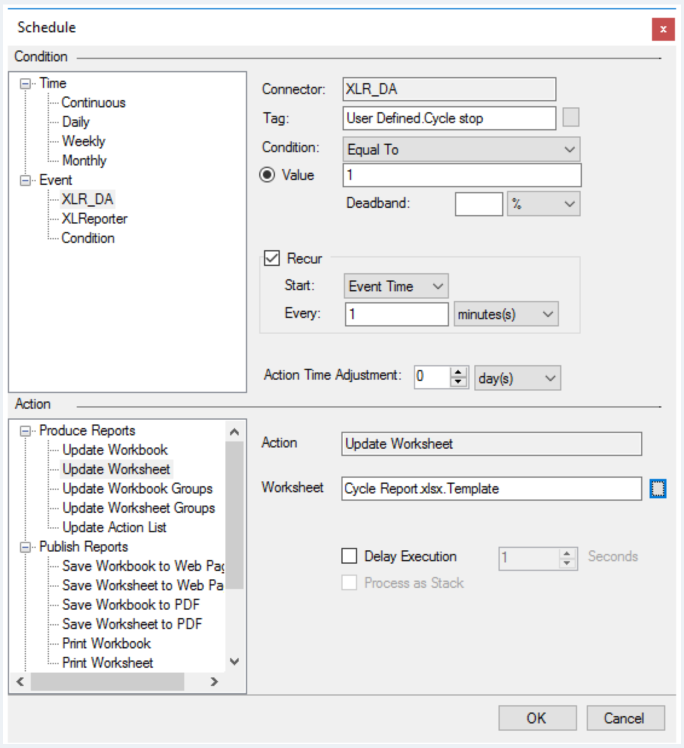
At the cycle start **Condition**, a **Set a Value to a Variable** action is used to set the value of the **Variable** *RG000* to a concatenation of the date and the value of the *Cycle ID* tag.



Connector: XLR_DA
Tag: User Defined.Cycle stop
Condition: Equal To
● Value: 1
Deadband: EU
☐ Recur
Start: Fixed Time 12:00:00 AM
Every: 1 minutes(s)
Action Time Adjustment: 0 day(s)

Action: Set a Value to a Variable
Variable: RG000
Value: {YYYY}_{MM}_{DD}_{<XLR DA>Grove Facility::Cyc
☐ Delay Execution 1 Seconds

This can be built by clicking the browse button […] for Value. The Calendar Item list provides keywords for year, month and day. The Connector Item branch opens a tag browser to select tags from real time connectors defined in the project.


Schedule

Condition

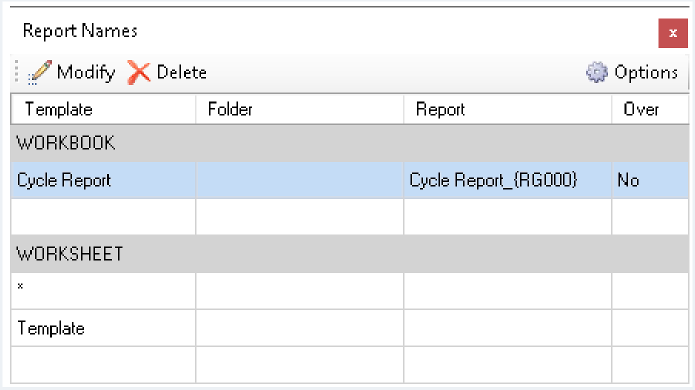
Connector: XLR_DA
Tag: User Defined.Cycle stop
Condition: Equal To
Value: 1
Deadband: %
Recur: ✓
Start: Event Time
Every: 1 minutes(s)
Action Time Adjustment: 0 day(s)

Action

Action: Update Worksheet
Worksheet: Cycle Report.xlsx.Template
Delay Execution: 1 Seconds
Process as Stack

The second action is configured with the same **Condition** as the first, but the **Recur** setting is enabled so that report is updated continuously while the cycle is running (Cycle start = 1).

Naming Convention


Report Names

Template	Folder	Report	Over
WORKBOOK			
Cycle Report		Cycle Report_{RG000}	No
WORKSHEET			
*			
Template			

The naming convention for this configuration sets the **WORKBOOK Name** to *{RG000}*. This allows the scheduler to generate a discrete report for each unique value of the variable, or in other words, each unique cycle ID combined with each unique date.

Data Placement

Like the previous scenario, the real time **Data Connection** is placed with the type *Insert at End, Down*.

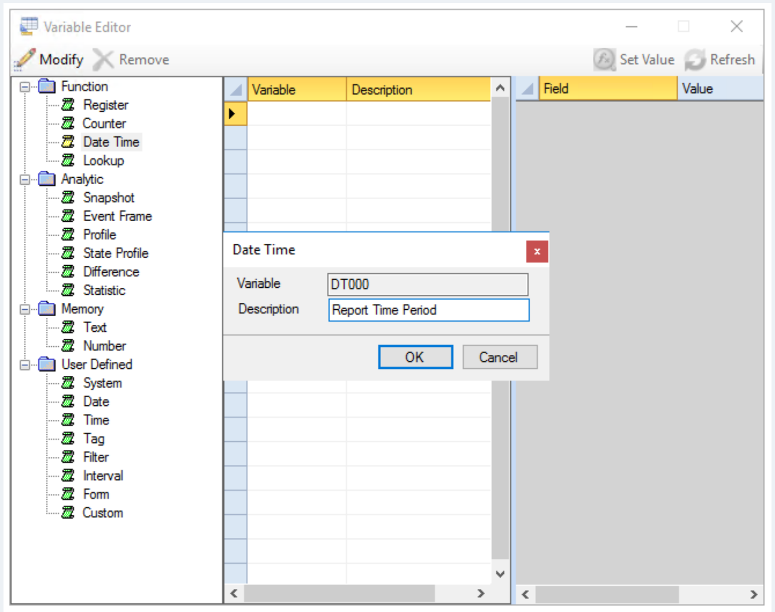
Batch History Report

In this scenario a report is required for each batch. The data for the batch report is stored in a continuous historian.

Variable

Like the previous example, this configuration uses a **Function, Register** variable in the naming convention, and it is set at the start of the cycle event.

This configuration also includes a **Function, DateTime** variable to track the start and stop of the event cycle for the purposes of querying the historical data.



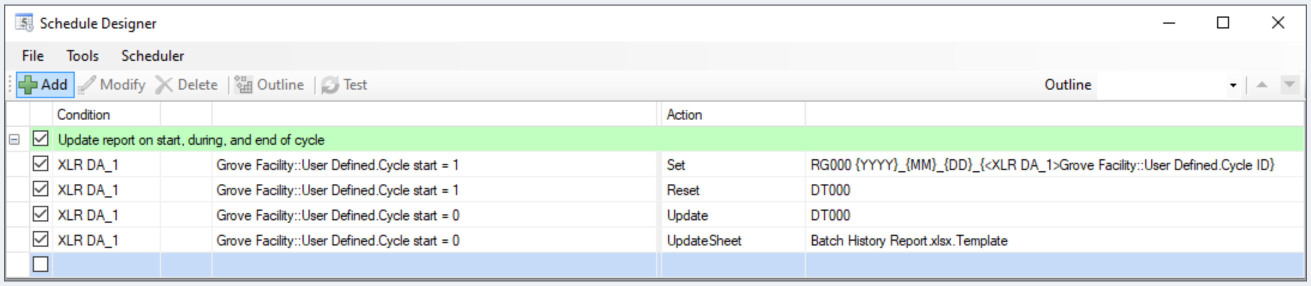
Date Time

Variable: DT000

Description: Report Time Period

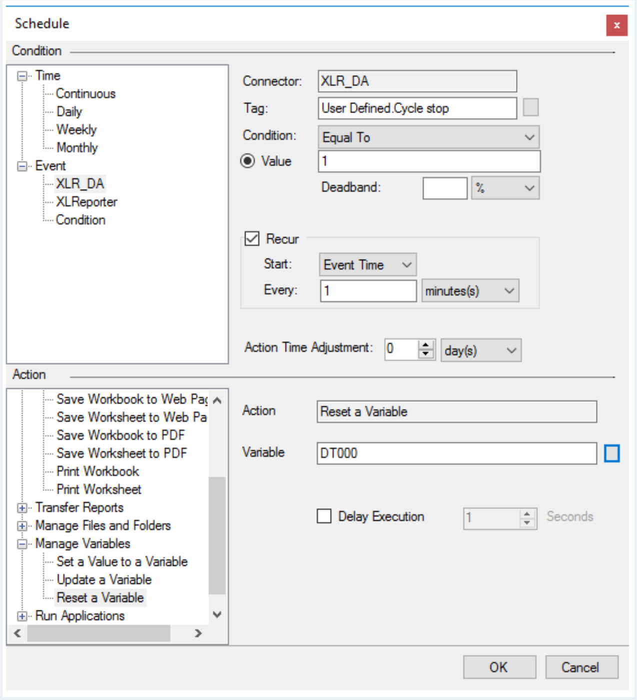
Schedule

The schedule includes four **Actions** in total. The first action, the value of *RG000* (the report name variable) is set to the value of the *Cycle ID* tag.



Condition			Action	
☑ Update report on start, during, and end of cycle				
☑ XLR DA_1		Grove Facility::User Defined.Cycle start = 1	Set	RG000 {YYYY}_{MM}_{DD}_{<XLR DA_1>Grove Facility::User Defined.Cycle ID}
☑ XLR DA_1		Grove Facility::User Defined.Cycle start = 1	Reset	DT000
☑ XLR DA_1		Grove Facility::User Defined.Cycle start = 0	Update	DT000
☑ XLR DA_1		Grove Facility::User Defined.Cycle start = 0	UpdateSheet	Batch History Report.xlsx.Template

In the second action, the *Reset* action is scheduled for the *DT000* variable. This sets the **Start Date** and **Start Time** fields of the variable to the time the cycle started.



Connector: XLR_DA

Tag: User Defined.Cycle stop

Condition: Equal To

Value: 1

Deadband: %

☑ Recur

Start: Event Time

Every: 1 minutes(s)

Action Time Adjustment: 0 day(s)

Action: Reset a Variable

Variable: DT000

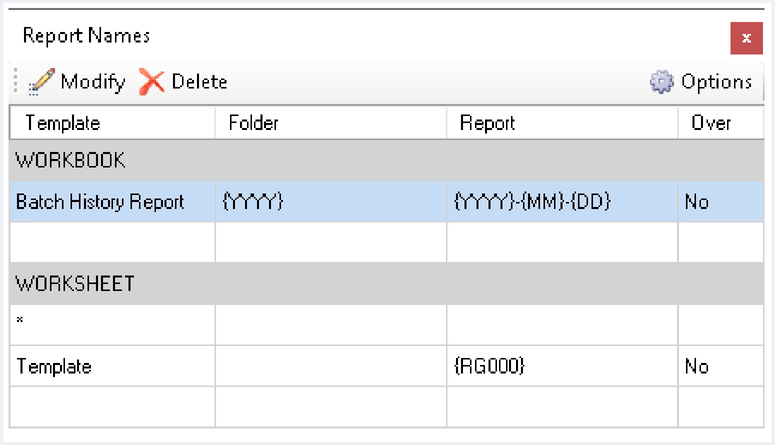
☐ Delay Execution 1 Seconds

The third action is processed at the end of the cycle event, in this case when *Cycle start* transitions to a value of *0*. This action runs the *Update* action on the *DT000* variable, which sets the **End Date** and **End Time** fields of the variable to the time the cycle ended.

Finally, the fourth action, also processed at the end of the cycle event, updates the *Batch History Report* template. Since the cycle ID, and its start and stop times have been established at this point, all the necessary information is available, and the report can be generated.

Naming Convention

This configuration expands on the naming convention discussed in the previous example. In that example, the *{RG000}* variable was set on the schedule to a concatenation of the cycle ID and the date. That way, a unique file would be generated based on a given cycle ID occurring on a given day.

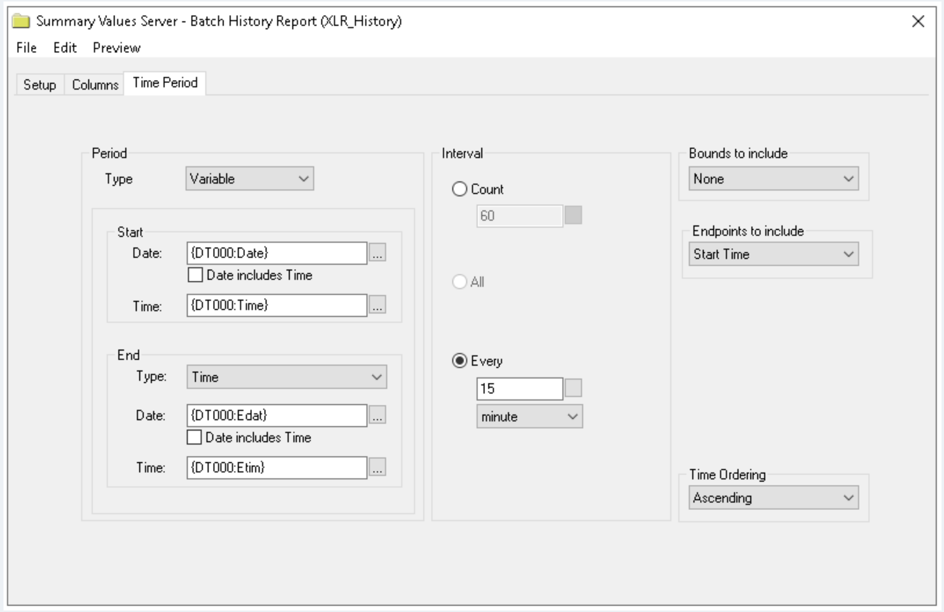


Template	Folder	Report	Over
WORKBOOK			
Batch History Report	{YYYY}	{YYYY}-{MM}-{DD}	No
WORKSHEET			
*			
Template		{RG000}	No

This configuration uses the **Folder** parameter to create a subfolder in the output based on the year. The **Workbook Name** is set to the year, month, and day. The **Template Sheet Name** is set to the variable *{RG000}*. So, the report for cycle *ABC123* which occurred on 3/20/2020 will be stored in [Project Report Path]\Batch History Report\2020\2020-03-20.xlsx. In a worksheet called *2020_03_20_ABC123*. If multiple cycles occurred that day, they would be placed in additional sheets in the same file. To instead store each cycle report in a different file, use a naming convention similar to the previous example.

Data Connection

Similarly, to both previous examples, the **Data Connection** in this template is placed using the *Insert at End, Down* **Placement Type**. However, because this template uses a **Historical Data Group** as the data source, the **Time Period** is defined based on the *DT000* variable.



Period
Type: Variable
Start
Date: {DT000:Date}
Date includes Time: ☐
Time: {DT000:Time}
End
Type: Time
Date: {DT000:Edat}
Date includes Time: ☐
Time: {DT000:Etim}

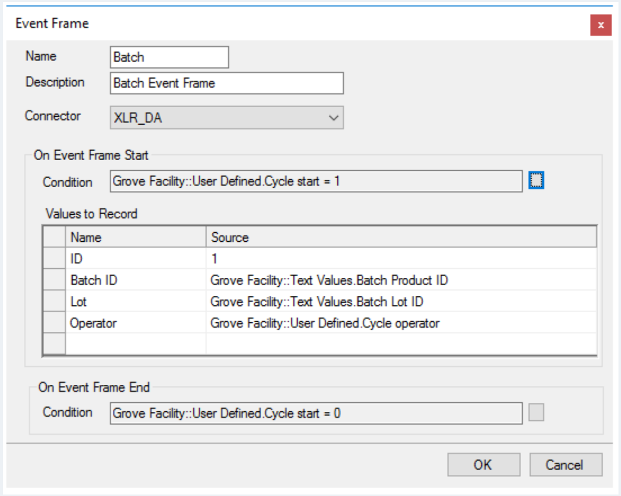
Interval
○ Count: 60
○ All
● Every: 15 minute

Bounds to include: None
Endpoints to include: Start Time
Time Ordering: Ascending

At the runtime of the report, which is triggered at the end of the cycle event, these variable fields resolve to the times that the event started and stopped respectively. The samples are collected every 15 minutes throughout the cycle based on the **Interval** parameter.

Batch History Report (Event Frames)

In this scenario a report is required for each batch. The data for the batch report is stored in a continuous historian. Note, this configuration requires the **Analytic Interface**. This configuration improves upon the previous example by removing the need for both **Registers** and **Date Time** variables and introduces **Event Frames**.



Name: Batch
Description: Batch Event Frame
Connector: XLR_DA

On Event Frame Start
Condition: Grove Facility::User Defined.Cycle start = 1

Values to Record

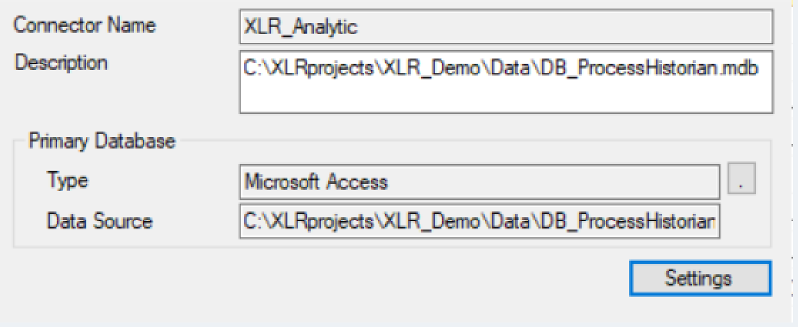
Name	Source
ID	1
Batch ID	Grove Facility::Text Values.Batch Product ID
Lot	Grove Facility::Text Values.Batch Lot ID
Operator	Grove Facility::User Defined.Cycle operator

On Event Frame End
Condition: Grove Facility::User Defined.Cycle start = 0

Event Frames not only simplify the configuration, but also provide the ability to reproduce past batch reports by logging information about each batch (including the start and end time) to a database.

Connector

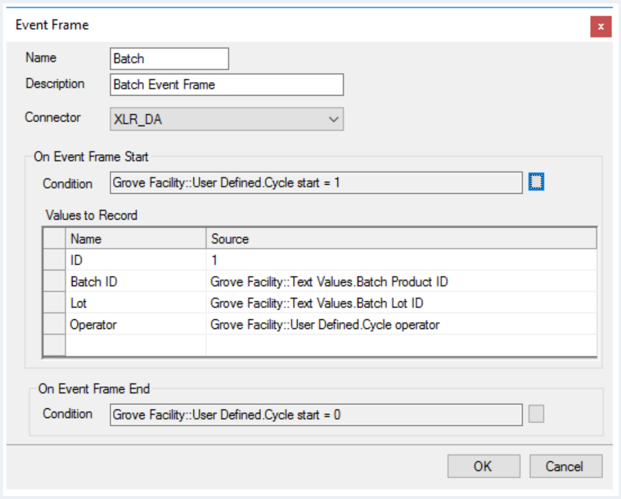
Before Event Frames can be configured, a **XLReporter, Analytic Values** connector must be set up.


Connector Name: XLR_Analytic
Description: C:\XLRprojects\XLR_Demo\Data\DB_ProcessHistorian.mdb

Primary Database
Type: Microsoft Access
Data Source: C:\XLRprojects\XLR_Demo\Data\DB_ProcessHistorian

The connector defines a connection to the database where the Event Frame data (as well as any other configured analytics) is stored.

Event Frame



Event Frame

Name	Batch
Description	Batch Event Frame
Connector	XLR_DA

On Event Frame Start

Condition: Grove Facility::User Defined.Cycle start = 1

Values to Record

Name	Source
ID	1
Batch ID	Grove Facility::Text Values.Batch Product ID
Lot	Grove Facility::Text Values.Batch Lot ID
Operator	Grove Facility::User Defined.Cycle operator

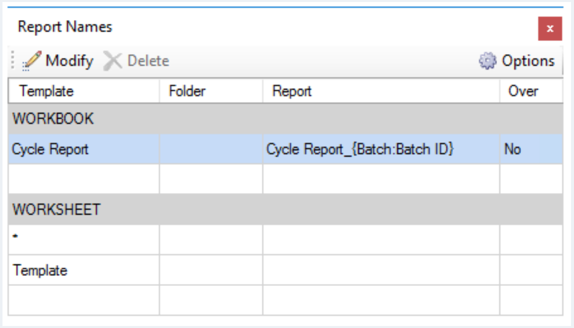
On Event Frame End

Condition: Grove Facility::User Defined.Cycle start = 0

The Event Frame is configured with Conditions that define when the batch starts and ends as well as process values that provide information about the batch. In this case, the *Batch ID*, *Lot* and *Operator* are recorded for each batch.

Naming Convention

This configuration further utilizes the Event Frame by naming the reports after the specified **Batch ID**.



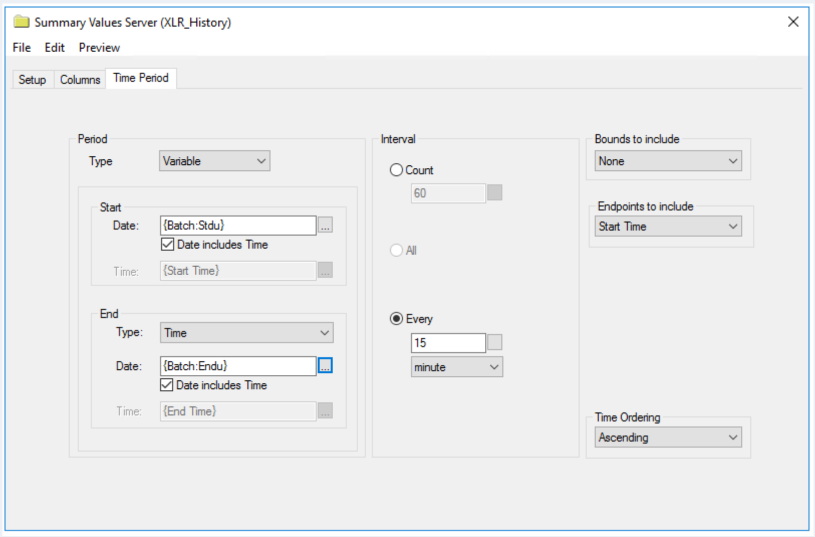
Report Names

Template	Folder	Report	Over
WORKBOOK			
Cycle Report		Cycle Report_{Batch:Batch ID}	No
WORKSHEET			
*			
Template			

This will result in a new report for each unique Batch ID.

Data Connection

Similarly, to the previous examples, the **Data Connection** in this template is placed using the *Insert at End, Down* **Placement Type**. However, because this template uses a **Historical Data Group** as the data source, the **Time Period** is defined based on the *Event Frame* variables.



Summary Values Server (XLR_History)

Period

Type: Variable

Start

Date: {Batch:Stdu}

☑ Date includes Time

Time: {Start Time}

End

Type: Time

Date: {Batch:Endu}

☑ Date includes Time

Time: {End Time}

Interval

○ Count: 60

○ All

● Every: 15 minute

Bounds to include: None

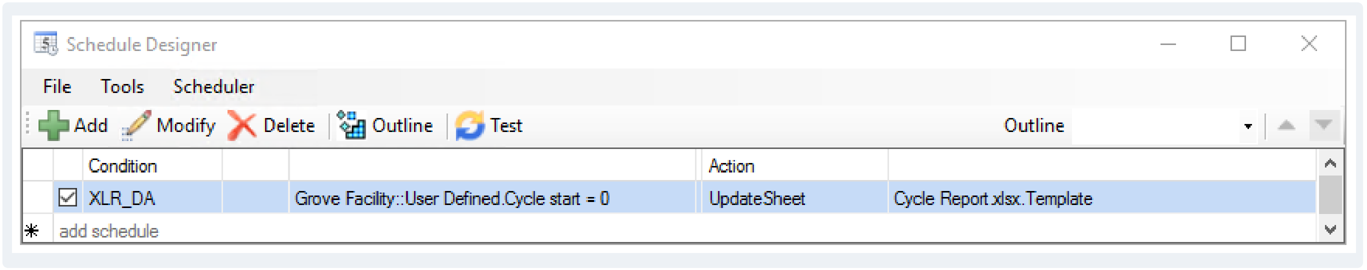
Endpoints to include: Start Time

Time Ordering: Ascending

At the runtime of the report, which is triggered at the end of the cycle event, these variable fields resolve to the times that the event started and stopped respectively. The samples are collected every 15 minutes throughout the cycle based on the **Interval** parameter.

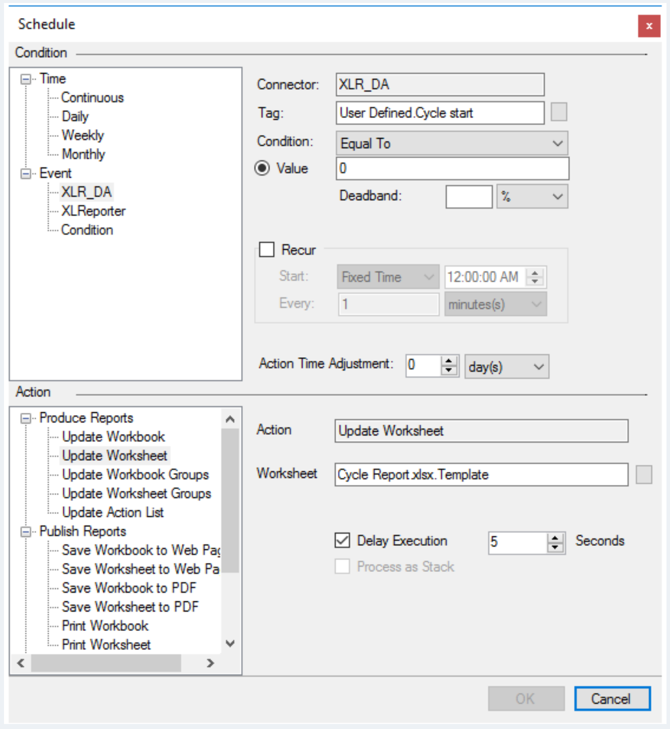
Schedule

The schedule includes a single **Action**.



	Condition			Action	
☑	XLR_DA		Grove Facility::User Defined.Cycle start = 0	UpdateSheet	Cycle Report.xlsx.Template
✳	add schedule				

In this action, the *UpdateSheet* action is scheduled for the *Cycle Report* template. This cause the report to be created at the end of the cycle. However, there is a slight difference between this update sheet and the one in the previous example.



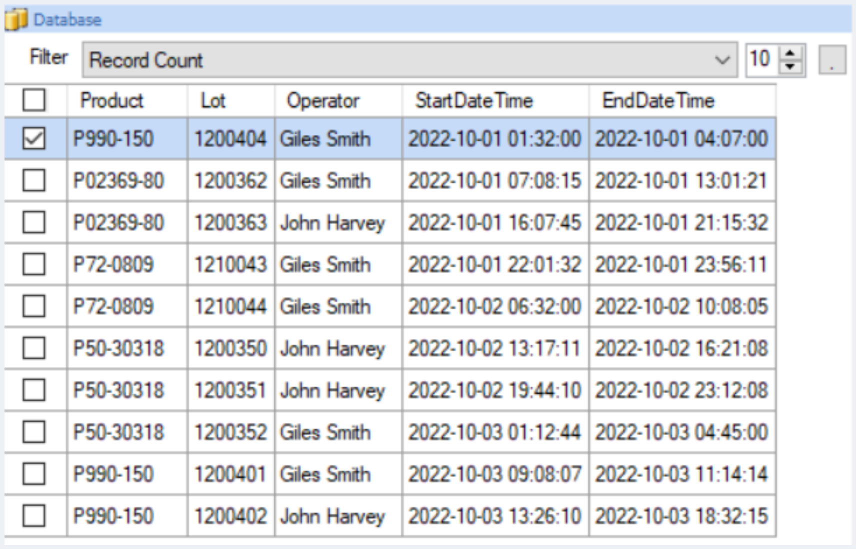
Connector: XLR_DA
Tag: User Defined.Cycle start
Condition: Equal To
● Value: 0
Deadband: %
☐ Recur
Start: Fixed Time 12:00:00 AM
Every: 1 minutes(s)
Action Time Adjustment: 0 day(s)

Action: Update Worksheet
Worksheet: Cycle Report.xlsx.Template
☑ Delay Execution 5 Seconds
☐ Process as Stack

Note that there is a 5 second **Delay Execution** configured. This allows the Event Frame a small window to store all required values. Since the *Batch ID*, as well as the start and stop times have been established at this point, all the necessary information is available, and the report can be generated.

On-Demand Reports

Because the Event Frame is configured to store a record for each batch, reports from this template can be generated On-Demand.



☐	Product	Lot	Operator	StartDateTime	EndDateTime
☑	P990-150	1200404	Giles Smith	2022-10-01 01:32:00	2022-10-01 04:07:00
☐	P02369-80	1200362	Giles Smith	2022-10-01 07:08:15	2022-10-01 13:01:21
☐	P02369-80	1200363	John Harvey	2022-10-01 16:07:45	2022-10-01 21:15:32
☐	P72-0809	1210043	Giles Smith	2022-10-01 22:01:32	2022-10-01 23:56:11
☐	P72-0809	1210044	Giles Smith	2022-10-02 06:32:00	2022-10-02 10:08:05
☐	P50-30318	1200350	John Harvey	2022-10-02 13:17:11	2022-10-02 16:21:08
☐	P50-30318	1200351	John Harvey	2022-10-02 19:44:10	2022-10-02 23:12:08
☐	P50-30318	1200352	Giles Smith	2022-10-03 01:12:44	2022-10-03 04:45:00
☐	P990-150	1200401	Giles Smith	2022-10-03 09:08:07	2022-10-03 11:14:14
☐	P990-150	1200402	John Harvey	2022-10-03 13:26:10	2022-10-03 18:32:15

When the template is selected, a list of batch records from the Event Frame is displayed allowing you to select the batch and generate the report for it.