Altair Engineering, Inc. makes no representation or warranties with respect to the contents of this manual or the associated software and especially disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Altair Engineering, Inc. reserves the right to revise this publication and make changes from time to time to its contents without obligation to notify anyone of such revisions or changes.

Altair Panopticon Streams is offered and is to be used in accordance with a SOFTWARE LICENSE AND MAINTENANCE AGREEMENT. This agreement stipulates that this software be used only in the computer system designated in that agreement. The agreement further stipulates that the customer shall not copy or alter, or permit others to copy or alter, the software or related materials in whole or in part, in any media for any purpose, except to make an archive (back-up) copy or to make a copy as an essential step in the use of the software with the customer's computer.

Altair Engineering, Inc. hereby grants the buyer the right to reprint this documentation for internal uses only. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, for any other purposes, without the prior written permission of Altair Engineering, Inc.

Altair Panopticon Streams, Reference Guide v17.4.0
Copyright © 2019 by Altair Engineering, Inc.
All rights reserved. Printed in the U.S.A.
Unpublished - Rights reserved under the copyright law of the United States.

Altair Panopticon Designer, Panopticon Designer, Altair Panopticon Streams and Altair Panopticon Server are trademarks of Altair Engineering, Inc. Other products mentioned herein may be trademarks or registered trademarks of their respective owners in the United States or other countries.

For U.S. Government End Users, the software is a "Commercial Item(s)," as that term is defined at 48 C.F.R. Section 2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 C.F.R. Section 12.212 or 48 C.F.R. Section 227.7202, as applicable. Consistent with 48 C.F.R. Section 12.212 or 48 C.F.R. Sections 227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the Altair Engineering, Inc. Software License and Maintenance Agreement.

SALES INFORMATION

US: + 1.800.445.3311
International: + 1.978.441.2200

Sales Email

US: sales@datawatch.com
Europe: sales_euro@datawatch.com
Asia Pacific: sales_apac@datawatch.com

SUPPORT CONTACT INFORMATION

Customer Portal: https://support.datawatch.com
Email: support@datawatch.com
US: +1 800.988.4739
International: +1 978.275.8350
[6] Using Altair Panopticon Streams ................................................................. 34

Connecting to or Disconnecting from the CEP Engine ........................................ 34
  Connecting to the CEP engine: ................................................................. 35
  Disconnecting from the CEP engine: ....................................................... 35


Uploading Applications .................................................................................. 36
  Sorting the List of Applications ............................................................... 39
  Searching for Applications ....................................................................... 40
Creating a New Application .......................................................................... 41
Adding an Operator ...................................................................................... 43
  Adding an Input Operator ........................................................................ 44
  Adding An Aggregation Operator ............................................................ 46
  Adding a Branch Operator ....................................................................... 53
  Adding a Calculation Operator ................................................................. 56
  Supported Operators and Calculation Functions ....................................... 58
  Adding a Conflate Operator ...................................................................... 61
  Adding an External Input ......................................................................... 63
  Adding a Filter Operator ......................................................................... 66
  Adding a Join Operator ........................................................................... 67
  Adding a Metronome Input Operator ........................................................ 74
  Adding a Python Transform Operator ....................................................... 76
  Adding a Rank Operator .......................................................................... 79
  Adding a Rekey Operator ........................................................................ 84
  Adding a Scatter Operator ...................................................................... 86
  Adding a Table to Stream Operator .......................................................... 88
  Adding a Union Operator ........................................................................ 91
  Adding an Output Operator ..................................................................... 92
Adding Application-specific Properties .......................................................... 94
  Saving an Application ............................................................................. 96
  Editing an Application ............................................................................ 97
  Validating and Fixing Application Issues ................................................ 98
Starting an Application .................................................................................. 99
  Starting an Application on the Applications Tab ..................................... 99
  Starting an Application on the Application page .................................... 101
Stopping an Application .............................................................................. 106
  Stopping an Application on the Applications Tab .................................. 106
  Stopping an Application on the Application Page .................................. 107
Viewing and Managing Application Topic Usages ....................................... 107
  Clearing the Data In an Application Topic ............................................. 108
  Clearing the Schema in an Application Topic ......................................... 108
  Sorting Application Topics .................................................................... 108
Other Applications Operations ..................................................................... 109
  Viewing Application History and Republishing .................................... 109
  Viewing the Application Data Sources Usage ....................................... 111
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downloading an Application</td>
<td>111</td>
</tr>
<tr>
<td>Deleting an Application</td>
<td>111</td>
</tr>
<tr>
<td>Moving to Other Applications Tab Pages</td>
<td>111</td>
</tr>
<tr>
<td>[8] Managing Data Sources</td>
<td>112</td>
</tr>
<tr>
<td>Uploading Data Sources</td>
<td>112</td>
</tr>
<tr>
<td>Creating a Data Source</td>
<td>115</td>
</tr>
<tr>
<td>Creating Email Output Connector</td>
<td>117</td>
</tr>
<tr>
<td>Creating InfluxDB Output Connector</td>
<td>118</td>
</tr>
<tr>
<td>Creating JDBC Database Output Connector</td>
<td>120</td>
</tr>
<tr>
<td>Creating Apache Kafka Output Connector</td>
<td>123</td>
</tr>
<tr>
<td>Creating Kx kdb+ Output Connector</td>
<td>126</td>
</tr>
<tr>
<td>Creating a Rest Output Connector</td>
<td>128</td>
</tr>
<tr>
<td>Creating Text Output Connector</td>
<td>130</td>
</tr>
<tr>
<td>Creating AMPS Input Data Source</td>
<td>132</td>
</tr>
<tr>
<td>Creating InfluxDB Input Data Source</td>
<td>135</td>
</tr>
<tr>
<td>Creating JDBC Database Input Data Source</td>
<td>139</td>
</tr>
<tr>
<td>Creating a JSON Input Data Source</td>
<td>140</td>
</tr>
<tr>
<td>Creating a Kafka Input Data Source</td>
<td>143</td>
</tr>
<tr>
<td>Creating Kx kdb+ Input Data Source</td>
<td>147</td>
</tr>
<tr>
<td>Creating Kx kdb+tick Input Data Source</td>
<td>148</td>
</tr>
<tr>
<td>Creating MQTT Input Data Source</td>
<td>151</td>
</tr>
<tr>
<td>Creating MS Excel Input Data Source</td>
<td>153</td>
</tr>
<tr>
<td>Creating OneTick Cloud Input Data Source</td>
<td>154</td>
</tr>
<tr>
<td>Creating RabbitMQ Input Data Source</td>
<td>156</td>
</tr>
<tr>
<td>Creating Solace Input Data Source</td>
<td>159</td>
</tr>
<tr>
<td>Creating Stream Simulator Input Data Source</td>
<td>162</td>
</tr>
<tr>
<td>Creating Text Input Data Source</td>
<td>165</td>
</tr>
<tr>
<td>Creating Websocket Input Data Source</td>
<td>168</td>
</tr>
<tr>
<td>Creating XML Input Data Source</td>
<td>171</td>
</tr>
<tr>
<td>Modifying Data Sources</td>
<td>175</td>
</tr>
<tr>
<td>Other Data Sources Operations</td>
<td>175</td>
</tr>
<tr>
<td>Sorting the List of Data Sources</td>
<td>176</td>
</tr>
<tr>
<td>Viewing the Application Usages</td>
<td>176</td>
</tr>
<tr>
<td>Downloading an Application Data Source</td>
<td>176</td>
</tr>
<tr>
<td>Deleting a Data Source</td>
<td>176</td>
</tr>
<tr>
<td>Searching for Data Sources</td>
<td>177</td>
</tr>
<tr>
<td>Refresh Data Producers</td>
<td>179</td>
</tr>
<tr>
<td>Starting or Stopping Data Producers</td>
<td>179</td>
</tr>
<tr>
<td>Managing Topics</td>
<td>182</td>
</tr>
</tbody>
</table>
[1] Introduction

Fundamental to understanding Panopticon Streams are these acronyms and terminologies:

**ACRONYMS**

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEP</td>
<td>Complex Event Processing</td>
</tr>
<tr>
<td>PCLI</td>
<td>Panopticon Command-line Interface</td>
</tr>
</tbody>
</table>

**TERMINOLOGY**

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Kafka or Kafka</td>
<td>Used for building the real-time data pipelines and streaming applications. It is horizontally scalable, fault-tolerant, fast and runs in production in thousands of companies.</td>
</tr>
<tr>
<td>Apache ZooKeeper or ZooKeeper</td>
<td>A centralized service for maintaining configuration information, naming, providing both distributed synchronization and group services.</td>
</tr>
<tr>
<td>Confluent</td>
<td>The free, open-source streaming platform based on Apache Kafka. The Confluent Platform is the complete streaming platform for large-scale distributed environments. Unlike a traditional messaging system or streaming processing API, Confluent Enterprise enables your interfaces to be connected to anywhere in the world and help make decisions with all your internal systems in real-time.</td>
</tr>
<tr>
<td>Schema registry</td>
<td>Part of the Confluent distribution package. Stores a versioned history of all schemas and allows the evolution of schemas according to the configured compatibility settings. Also provides a plug-in to clients that handles schema storage and retrieval for messages that are sent in Avro format.</td>
</tr>
<tr>
<td>Panopticon Streams</td>
<td>The name of the Panopticon CEP platform.</td>
</tr>
</tbody>
</table>
Overview

Event processing is a method of tracking and analyzing streams of information of an event, and eventually deriving a conclusion from what transpired. CEP is an event processing method which combines data from multiple sources to infer events or patterns that may demonstrate unusual activities or anomalies, consequently requiring immediate action.

The CEP engine provided by Panopticon is named Panopticon Streams and it is built to work with different CEP engines. However, for this version, it will only support Kafka.

Kafka is a distributed streaming platform which lets you publish and subscribe to streams of records. Each record consists of a key, a value, and a timestamp and stores streams of records in categories called topics. Kafka is mainly used for two reasons:

- Building real-time streaming data pipelines that reliably get data between systems or applications
- Building real-time streaming applications that transform or react to the streams of the data

Refer to https://kafka.apache.org/intro.html for more information.

Panopticon Streams enables you to create streaming data pipelines which both transforms and reacts to streaming data. Aside from Kafka, it is also using ZooKeeper and Schema Registry provided by Confluent. ZooKeeper is a key component when using Kafka since it allows the configuration and management of clusters in the Kafka servers. The Schema Registry stores a versioned history of all schemas used by Kafka and provides a RESTful interface for storing and retrieving Avro schemas.

PANOPTICON STREAMS APPLICATIONS

The main task of the Panopticon Streams is to execute and manage streams applications. An application describes how data should be piped, transformed, and processed. Applications consist of a set of inputs, operators, and outputs and is described or constructed in an XML file.

It can be viewed as a directed graph with a set of nodes (or operators) and a set of edges (or streams) that are interconnected with each other.

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>The ID of the application config. It should be the same with the filename when loading an application config from the system.</td>
</tr>
<tr>
<td>operators</td>
<td>A list of operators (actions and functions).</td>
</tr>
<tr>
<td>streams</td>
<td>A list of streams that describe the connection and the flow between operators.</td>
</tr>
<tr>
<td>properties</td>
<td>Application-specific defined properties.</td>
</tr>
</tbody>
</table>
An application can either use **Kafka topics** or **data producers**, which generate data from a data source. The data producer also demonstrates to be the connection between the Panopticon Streams framework and the Panopticon core.

The Panopticon core has data connectors such as Kx kdb+, OneTick, and MS Excel that serve as data sources. Just like the application, the data source is also constructed or described in an XML file.

**NOTES**

The current standalone Panopticon Streams application include the following data producers:

**Data Producers:**

- ActiveMQ
- AMPS
- InfluxDB
- JDBC Database
- JSON
- Kafka
- Kx kdb+
- Kx kdb+ Tick
- MQTT
- MS Excel
- RabbitMQ
- Solace
- Stream Simulator
- Text
- WebSocket
- XML

An application refers to a data source through its ID (or filename). There are several ways to create a data source of an application:

- Export data sources in the **Panopticon Designer**
- Export data source with the **PCLI tool**

  The PCLI tool extracts the already defined data sources in workbooks and saves them as CEP data sources.
PANOPTICON STREAMS OPERATORS

An **operator** is a single task responsible for processing the data and publishing it as an output. Currently, the Panopticon Streams supports 14 operators:

- Aggregation
- Branch
- Calculation
- Conflate
- External Input
- Filter
- Input
- Join
- Metronome
- Rank
- Rekey
- Scatter
- To_stream
- Output
- Union
- Python Transform

Each operator produces one or more output streams that can be connected and defined as input streams for other operators.

Panopticon Streams Inputs

The Panopticon Streams engine allows the combination of multiple data sources and their definition as input channels. The data sources are referred to within the Panopticon Streams as **inputs**. The data produced by each input can be processed by one or more operators.

Panopticon Streams Outputs

An **output** produces and publishes streams towards a Kafka topic or a **data consumer**. A data consumer is the opposite of a data producer. It consumes the data produced from an output in Panopticon Streams and publishes the data to a data source.

The most common approach is to publish the data to a Kafka topic which eventually can be consumed or used by the Panopticon Designer, Panopticon Server, or other platforms that support Kafka.

Currently, Panopticon Streams supports publishing of the output data to the following data sources:

- Email
- InfluxDB
- JDBC Databases
- Apache Kafka
- Kx kdb+
- Rest
- Text

This section discusses the steps in setting up the ZooKeeper, Kafka, Schema Registry, and the Panopticon Streams.

Before proceeding, you must install and setup the following prerequisites:

 Java JDK 64-bit
 System Environment variable JAVA_HOME set to the Java JDK 64-bit

Setting Up ZooKeeper, Kafka, and Schema Registry

Steps:

1. Download the packaged file.
2. Extract the contents of the Streams.zip file to a new location (e.g., C:\). This zip file will contain the following folders and files:
   - Confluent
   - Extra
   - ZooKeeper
   - 1-start-zookeeper.bat
   - 2-start-kafka.bat
   - 3-start-schema-registry.bat
   - INSTRUCTIONS.txt
   - VERSIONS.txt

3. Run the batch files in the correct order. Make sure the previous batch file has been started before continuing to the next one:
   - 1-start-zookeeper.bat
   - 2-start-kafka.bat
   - 3-start-schema-registry.bat

When these three batch files have been started, you can now connect to the Standalone Panopticon Streams to execute and deploy your applications.
NOTES

When connecting to a Kafka broker on a separate machine, exposing different IP addresses internally and externally, you need to configure KAFKA_ADVERTISED_LISTENERS. This is typically the case when running Kafka in a Docker container.

The symptoms of the missing configuration are:

- Panopticon Streams can connect to ZooKeeper and the Kafka Broker
- No data is written to topics

In [Kafka]/etc/kafka/server.properties, uncomment advertised.listeners and replace "your.host.name" with the externally exposed host name or IP address.

```plaintext
# Hostname and port the broker will advertise to producers and consumers. If not set, # it uses the value for "listeners" if configured. Otherwise, it will use the value # returned from java.net.InetAddress.getCanonicalHostName(). advertised.listeners=PLAINTEXT://your.host.name:9092
```

When using the Confluent Docker image, you can pass the KAFKA_ADVERTISED_LISTENERS as a parameter:

```bash
docker run -d --restart=always \
  --net=confluent \
  --name=kafka \
  -p 9092:9092 \
  -e KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181 \
  -e KAFKA_ADVERTISED_LISTENERS=PLAINTEXT://your.host.name:9092 \
  -e KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR=1 \
  confluentinc/cp-kafka:5.1.0
```

ADDITIONAL NOTES ON SETTING UP THE SCHEMA REGISTRY

It is recommended to turn off the compatibility checking in schema registry when used with Panopticon Streams.

To do this, set the Avro compatibility level to NONE (as mentioned below) in the schema-registry.properties file.

Then there are three cases depending on how Kafka is deployed:

- On Windows from the ZIP file from Panopticon. Already turned off by default.
On Linux manually deployed ("bare metal"). Add the following line to ...etc\schema-registry\schema-registry.properties
Avro.compatibility.level=NONE

With Docker Compose using the Confluent images
Add the following line to the environment section of the schema-registry service in docker-compose.yml
   SCHEMA_REGISTRY_AVRO_COMPATIBILITY_LEVEL: 'NONE'

Setting Up Panopticon Streams

NOTES

- The standalone Panopticon Streams application requires ZooKeper, Kafka, and Schema Registry.
- If you need to upgrade your previously installed Panopticon Streams, proceed to the Upgrade section.

Steps:

1. Extract the contents of the PanopticonStreamsWAR_<version>.zip file to a new location.
   This zip file will contain the following files:
   - streams.war
   - streams.xml

2. Create the AppData folder (i.e., streamsserverdata) and ensure that the user account Local Service running Tomcat has read/write and execute permissions to this folder.
   Example: c:\streamsserverdata

3. Copy the extracted streams.xml file into the Tomcat config folder (\Apache Software Foundation\Tomcat 9.0\conf\Catalina\localhost). This file contains the following information:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Context path="/streams">
   <Environment name="DatawatchVDDAppData" override="false" type="java.lang.String" value="c:\streamsserverdata" />
</Context>
```
NOTES

Update this file if the path of the environment variable DatawatchVDDAppData is different from c:\streamssserverdata or the data folder created in step 2.

4. Copy the streams.war file into the Tomcat webapps folder (\Apache Software Foundation\Tomcat 9.0\webapps).

5. Edit the existing tomcat-users.xml file which is available in the Tomcat config folder (\Apache Software Foundation\Tomcat 9.0\conf) and add the entry:

   <user username="designer" password="designer" roles="user"/>

   For more complex authentication and user directory options, see section [3] Authentication.

6. Start Tomcat to deploy the .war file.

   The streams folder is extracted in the Tomcat webapps folder:

   ![File List](image)

   Also, the CEP and Data folders are generated in the streamssserverdata folder along with the streams.properties file:

   ![File List](image)

7. Specify the license type that will be used. Refer to Licensing for more information.

8. You should now be able to log on to the Panopticon Streams using the following:

   [Host Name]:[Port]/[Name of your application]

   For example:

   http://localhost:8080/streams

Reference Guide 17.4.0 9
Licensing

Licensing within the Panopticon Streams supports two license types:

- a volume-based XML file (named **DatawatchLicense.xml**) which is used to store all license information for a specific customer
  
The Panopticon Streams allows the systems administrator to copy the license file to the designated **AppData** folder (i.e., **streamsserverdata**).

- **HyperWorks Units license** which is available in the Altair License Server you are connected to (local or over the network)

The license file type you will use is delivered separately from the installation packages.

### USING HYPERWORKS UNITS LICENSE IN THE PANOPTICON STREAMS

Before using the HyperWorks Units license type in the Panopticon Streams, it is required to configure certain properties in the **Streams.properties** file located in the **AppData** folder or **c:\streamsserverdata**:  

<table>
<thead>
<tr>
<th>Property</th>
<th>Attribute</th>
<th>Description</th>
<th>Default Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Service authentication level</td>
<td>The property that will make the authentication</td>
<td>false</td>
<td>Multiple License Servers are not supported when the Panopticon Stream is on a Linux machine.</td>
</tr>
<tr>
<td>Property</td>
<td>authentication.required</td>
<td>required. It will force the user to login in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>order to use any of the services</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>provided by the server. Must be set to true.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Licensing</td>
<td>The operating system where the Panopticon Streams</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>license.hwu.operating.system</td>
<td>is installed. Possible values are: <strong>WIN_X86</strong>, <strong>WIN_X64</strong>, <strong>MAC</strong>, <strong>LINUX_X64</strong>, or <strong>LINUX_ARM64</strong>.</td>
<td></td>
<td>Note: If the Java bitness (e.g., 32-bit) is different from the operating system (e.g., 64-bit), it is recommended to add the Java bitness in this property (e.g., WIN_X86).</td>
</tr>
<tr>
<td>Property</td>
<td></td>
<td>The path where the License Server is running</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>license.hwu.uri</td>
<td>e.g., <strong>6200@191.255.255.0</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td>where the syntax is PORTNUMBER@HOST. If multiple</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>servers are used, they should be separated by ';'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
<td>Notes:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Multiple License Servers are not supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>when the Panopticon Stream is on a Linux machine.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default Value</td>
<td>Property</td>
<td>Licensing</td>
<td>Attribute</td>
<td>licensing.hwu.version</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>19.0</strong></td>
<td><strong>FILE</strong></td>
<td><strong>WIN_X64</strong></td>
<td><strong>WIN_X64</strong></td>
<td><strong>WIN_X64</strong></td>
</tr>
</tbody>
</table>

For example:

authentication.required=true
license.hwu.operating.system=WIN_X64
license.hwu.uri=6200@192.168.5.51;6200@192.168.5.52
license.hwu.version=19.0
license.mode=HWU

**NOTES**

- The Panopticon Streams doesn't consume licenses for user login
- Each application consumes 10000 licenses while in running state
- Each scheduler consumes 2000 licenses while in running state

**Upgrade**

A previously installed Panopticon Streams can be upgraded through the following process:

1. Stop Tomcat.
2. Delete the existing webapps\streams.war file.
3. Delete the deployed application: webapps\streams
4. Delete the cache from the working folder (for example):
   work\Catalina\localhost\streams
5. Deploy the new `streams.war` file by copying it to the Tomcat `webapps` folder.

6. Restart Tomcat.

Introduction

The Panopticon Streams provides multiple approaches on authentication. It can easily be configured to use different authentication mechanisms depending on the environment and the setup. The server only supports authentication and authorization and does not have any support for user management or administration of users.

There are mainly two properties that manage the authentication on the server. These properties are listed and described in the table below. Please note that more properties might need to be configured depending on the authentication mechanism you are using.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication.role</td>
<td>The required role or group that the user needs to be identified as a Panopticon user. The property can be left blank if no role or group is required.</td>
<td>User</td>
</tr>
<tr>
<td>authentication.required</td>
<td>This property will make the authentication required. It will force the user to login in order to use any of the services provided by the server.</td>
<td>false</td>
</tr>
<tr>
<td>authentication.type</td>
<td>The type of authentication that should be used when authenticating the user. The property allows the following values: BASIC, FILTER, HEADER, SAML, WINDOWS.</td>
<td>BASIC</td>
</tr>
</tbody>
</table>

The web user interface supports all of the authentication mechanisms that are listed in this chapter. However, the Panopticon Designer only supports certain authentication mechanisms such as listed below:

- Tomcat Realm
- LDAP
- Active Directory
- Windows

Refer to the sections below for more information.

TOKEN

A web token is used when the user has successfully logged into the Panopticon Streams when using one of the following authentication types: BASIC, SAML, or WINDOWS. The token is used to identify the user and represent the user's ongoing session. This is done to prevent user credentials being sent between the user and server more than necessary.

The token is returned from the Panopticon Streams in the form of a cookie when the user has been authenticated. The cookie will be stored in the browser as a HttpOnly cookie.
The token can be configured differently to suit your needs and requirement. The token can be configured to be valid at a certain amount of time, if it can refresh itself and/or if it should be persistent or if it should only last for a user session (While the browser is still open). All this can be configured in the `streams.properties`. The table below lists all available token properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication.token.persistence</td>
<td>This property is used to determine if the token should persist if the browser is closed or if it should only last while the browser is open. There are two possible values: PERSISTENT and SESSION. PERSISTENT will persist the token in the browser even if the browser has been closed and reopened. SESSION will remove the token from the browser if it is shutdown.</td>
<td>PERSISTENT</td>
</tr>
<tr>
<td>authentication.token.refreshable</td>
<td>This property determines if the token can refresh itself. The web client can identify if the token is about to expire and then request a new token with the existing token. A token is refreshable if the property is set to true. The token will expire and invalidate the user session if the property is set to false.</td>
<td>true</td>
</tr>
<tr>
<td>authentication.token.refreshable.scope</td>
<td>This property determines who can refresh a token: ALL or CLIENT. ALL means that both the client and the internal subscriptions can refresh a token for a user. This ensures that an internal subscription will always be authenticated. CLIENT means that only the client can refresh the token. This prohibits the server from refreshing a token for an internal subscription. Therefore, it is beneficial in ensuring a user interaction is required to refresh the token. This is recommended when there are more security constraints and a short lifespan on the token.</td>
<td>ALL</td>
</tr>
<tr>
<td>authentication.token.secret</td>
<td>The secret is used to sign the token. The secret will be auto-generated when the server starts for the first time. <strong>NOTE: This value should be kept a secret.</strong></td>
<td>Auto-generated</td>
</tr>
<tr>
<td>authentication.token.validity.seconds</td>
<td>The number of seconds that the token should be valid.</td>
<td>604800</td>
</tr>
</tbody>
</table>
Tomcat Realm

The Panopticon Streams can be configured to use the Tomcat Realm when performing authentication. The Tomcat Realm is configured in the `server.xml` file in the Tomcat `conf` folder. The Tomcat Realm itself can be configured to authenticate towards a variety of different types of authentication source, such as Tomcat user base and LDAP. The sub chapters in this chapter will give examples on how to configure the Tomcat Realm.

The Panopticon Streams needs to be configured to use the BASIC type in order to do the authentication towards the Tomcat Realm. To enable Tomcat Realm authentication, set this property in the `streams.properties` file:

```
authentication.type=BASIC
```

**NOTES**

It is a common approach to wrap your Tomcat Realm with the LockOutRealm. This is used to prevent brute-force attacks.

```
<Realm className="org.apache.catalina.realm.LockOutRealm">
  <!--Insert your own Tomcat Realm here -->
</Realm>
```

**TOMCAT USER BASE**

The Tomcat User Base Realm is using a JNDI resource to store user information. By default, the JNDI resource is configured in an XML file. The default file is `tomcat-users.xml` in the Apache Tomcat `conf` folder.

We strongly recommend using this authentication approach for your test or local environment. It is easy to setup and configure. However, it is not designed to be used for large-scale production or when you have a large number of users.

The following Realm has to be added in the `server.xml` file in the Apache Tomcat `conf` folder:

```
<Realm className="org.apache.catalina.realm.UserDatabaseRealm"
       resourceName="UserDatabase"/>
```

**NOTES**

The Tomcat User Database Realm is used as the default. No configurations are required in the `server.xml` file to be able to use the Tomcat Database Realm.
The users and roles are managed in the `tomcat-users.xml` file in the Apache Tomcat `conf` folder. In this file, you can add users and roles as well as assign roles to users.

Add the following role and user to your `tomcat-users.xml` file:

```xml
<role rolename="user"/>
<user username="designer" password="designer" roles="user"/>
```

By adding these two lines you have achieved the following:

- Created a new role named `user`
- Created a new user with username `designer` and password `designer`
- Assigned the newly created user the role `user`

**NOTES**

- Authentication towards a Tomcat Realm (i.e., Tomcat users, LDAP, AD) in Tomcat 8.5.28 is not supported. This has been supported in all the previous and the succeeding versions.
- A sample `tomcat-users_example.xml` is provided in the `DatawatchVisualizationServerWAR_<version number>.zip` file.

**Tomcat Memory Configuration**

**NOTES**

It is recommended to increase the Java heap size of Tomcat to avoid the initiation of garbage collection when memory usage hits the set threshold.

The steps may vary depending on how Tomcat was deployed.

**Steps:**

1. Stop Tomcat.
2. You can either create a file named `setenv.bat` (for Windows) or `setenv.sh` (for Linux).
3. Place them in the Tomcat `bin` folder.
4. Set the minimum and maximum heap size with the JVM `-Xms` and `-Xmx` parameters. A minimum of 1 GB is recommended. For example:
   - **For Windows:**
     ```
     set JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8 -server -Xms512m -Xmx2g
     ```
   - **For Linux:**
     ```
     JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -server -Xms512m -Xmx2g"
     ```
NOTES
Setting the maximum value should be dependent on your system. Ensure that the heap size is not larger than the available free RAM on your system. It is recommended to use 80% of the available RAM not taken by the operating system or other processes of your JVM.

5. Set the maxim limit of the PermGen (Permanent Generation) memory space to a size larger than the default. The default value (64 MB) is not enough for enterprise applications. 256 MB is recommended in most cases. You can set the PermGen maximum limit with the following JVM parameter:
   -XX:MaxPermSize=256m

NOTES
If the PermGen space maximum is too low, OutOfMemoryError: PermGen space errors may occur.

6. Save the file.
7. Restart the Tomcat service to apply the increase in the heap size and the new Permgen maximum limit.

Another option in setting the heap size is through the System Variables. Follow these steps:

1. Stop Tomcat.
2. Go to System Environment variables. (Right-click Computer > Properties > Advanced System Parameters > Environment Variables.)
   The System Properties dialog displays.
3. Select the Advanced tab and click Environment Variables.
4. Click New under the System Variables section.
5. Define the following:
   • Variable Name: CATALINA_OPTS
   • Variable value: -Xms512m -Xmx2g
6. Click OK then OK again.
7. Restart the Tomcat service.
LDAP

The Panopticon Streams can be configured to authenticate towards a Lightweight Directory Access Protocol (LDAP) or source. By configuring the Apache Tomcat Realm, the server can authenticate users and extract their roles by querying the LDAP source.

The realm’s connection to the directory is defined by the `connectionURL` attribute. Each user that can be authenticated must be represented in the directory with an individual entry that corresponds to an element in the initial `DirContext` from the `connectionURL`. This user entry must have an attribute containing the username that is presented for authentication.

You can add a dedicated user with `connectionName` and `connectionPassword` in a Realm to define a user with a Read access to the user database and roles. If for example the admin `cn` name is set as `admin` and the admin `password` is set as `admin`, then you need to add these properties as shown in the example below.

The `userPattern` attribute may be used to specify the DN, with “{0}” marking where the username should be substituted.

The role is usually an LDAP group entry with one attribute containing the name of the role and another one whose values are distinguished names or usernames of the users in that role. The following attributes configure a directory search to find the names of roles associated with the authenticated user:

- **roleBase**: The base entry for the role search. If not specified, the search base is the top-level directory context.
- **roleSearch**: The LDAP search filter for selecting role entries.
- **roleName**: The attribute in a role entry containing the name of that role.
- **roleNested**: Includes nested roles if set to `true`. This means every newly found `roleName` and distinguished Name will be recursively tried for a new role search. The default behavior is `false`.

The following is an example on how the Realm can be configured when using LDAP. Please note that the values should be replaced with details from your own LDAP source.

```xml
<Realm className="org.apache.catalina.realm.JNDIRealm"
    connectionURL="ldap://localhost:389"
    connectionName="cn=admin,dc=test,dc=com"
    connectionPassword="admin"
    userPattern="uid={0},ou=users,dc=test,dc=com"
    roleBase="ou=groups,dc=test,dc=com"
    roleName="cn"
    roleSearch="(uniqueMember={0})"
    rolenested="true"
/>
```

Using this configuration, the realm determines the user’s distinguished name by substituting the username into the `userPattern`, authenticates by binding to the directory with this DN and the password received from the user, and searches the directory to find the user’s roles.
NOTES

If you opt not to have a dedicated user, remove `connectionName` and `connectionPassword`, and then have each user extract information about itself. You do this by adding `userSearchAsUser` and `roleSearchAsUser` in a Realm, and setting both values to `true`. The recommended usage, however, is to have a dedicated user. This allows you to always have the rights to query a LDAP, unlike using `userSearchAsUser` and `roleSearchAsUser` where there is no guarantee that each user is authorized to extract these details.

ACTIVE DIRECTORY

The Panopticon Streams can be configured to authenticate towards an Active Directory server. The Panopticon Streams is using LDAP to interact and communicate with the Active Directory server. Therefore, the configuration is very similar to the LDAP configuration in the previous section.

The following is an example on how the Realm can be configured when using Active Directory. Please note that the values should be replaced with details from your own LDAP source.

```xml
<Realm className="org.apache.catalina.realm.JNDIRealm"
    connectionURL="ldap://ad.dwch.com:3268"
    alternateURL="ldap://ad.dwch.com:389"
    authentication="simple"
    referrals="follow"
    connectionName=admin@DWCH.com
    connectionPassword="admin"
    userBase="cn=Users,dc=DWCH,dc=com"
    userSearch="(sAMAccountName={0})"
    userSubtree="true"
    roleBase="cn=Users,dc=DWCH,dc=com"
    roleName="cn"
    roleSearch="(member={0})"
    roleSubtree="true"
    roleNested="true"
/>
```
NOTES

Similar with LDAP, you can opt not to have a dedicated user by removing `connectionName` and `connectionPassword` and instead let each user extract information about itself by adding `userSearchAsUser` and `roleSearchAsUser` in a Realm. Set both values to `true`. As mentioned in the LDAP section, the recommended usage is to have a dedicated user since there is no guarantee that each user is authorized to extract these details.

**Windows Authentication**

The Panopticon Streams supports Windows authentication. The Panopticon Streams will authenticate a user towards the local machine and verify its credentials with the existing and configured users on the Windows machine. The Windows authentication operates similarly to the Basic authentication function. Both the username and the password are sent to the Panopticon Streams which they are then verified.

To enable Windows authentication, set this property in the `streams.properties` file:

```
authentication.type=WINDOWS
```

NOTES

Single Sign On is currently not supported with the Windows authentication. In addition, Windows authentication only supports authentication towards the local machine. This means that the machine where the Panopticon Streams is deployed on also has to manage all of the users.

**SAML**

The Panopticon Streams supports Security Assertion Markup Language, SAML2. Upon a login request, the Panopticon Streams will redirect the user to an Identity provider (IdP). The IdP will authenticate the user and redirect the user back to the Panopticon Stream. The response message will be controlled and validated. Username and roles will be extracted from the response message and used within the Panopticon Streams.

The Panopticon Streams will redirect the user back to the IdP upon a logout request. The IdP logout service should then invalidate the SAML token.
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication.saml.assertion.roles</td>
<td>User attribute for roles configured in the IdP.</td>
</tr>
<tr>
<td>authentication.saml.assertion.username</td>
<td>User attribute for username configured in the IdP.</td>
</tr>
<tr>
<td>authentication.saml.assertionservice.url</td>
<td>The URL to the Panopticon assertion consumer service. URL: [Protocol]://[Host]:[Port]/[Context]/serve r/rest/auth/login Example: <a href="http://localhost:8080/panopticon/server/rest/auth/login">http://localhost:8080/panopticon/server/rest/auth/login</a></td>
</tr>
<tr>
<td>authentication.saml.callback.url</td>
<td>Relay state.</td>
</tr>
<tr>
<td>authentication.saml.certificate.name</td>
<td>The name of the certificate used to validate signature.</td>
</tr>
<tr>
<td>authentication.saml.certificate.password</td>
<td>The password of the certificate used to validate signature.</td>
</tr>
<tr>
<td>authentication.saml.challenge.required</td>
<td>Determines whether the IdP-first authentication with SAML is enabled or not. To enable, set this property to false.</td>
</tr>
<tr>
<td>authentication.saml.identityprovider.logo ut.url</td>
<td>The URL to the IdP logout service.</td>
</tr>
<tr>
<td>authentication.saml.identityprovider.url</td>
<td>The URL to the IdP login service.</td>
</tr>
<tr>
<td>authentication.saml.keystore.file</td>
<td>The location of the Keystore file that contains the certificate.</td>
</tr>
<tr>
<td>authentication.saml.keystore.password</td>
<td>The password to the Keystore file.</td>
</tr>
<tr>
<td>authentication.saml.redirect</td>
<td>Redirects the user back to the Panopticon Streams URL. This is mainly used with a proxy. In which case, the Panopticon Streams does not know the endpoint which the user is going towards to, and therefore cannot redirect the user back to the Overview page. This can be left blank.</td>
</tr>
<tr>
<td>authentication.saml.serviceprovider.id</td>
<td>The ID of the service provider configured in the IdP.</td>
</tr>
</tbody>
</table>

### OAuth 2.0

This section discusses how to configure the Panopticon Streams to use the OAuth 2.0 for authorization. Upon a logon request, the Panopticon Streams will redirect the user to the Login page provided by the OAuth 2.0.

Note that OAuth 2.0 does not normally provide support on how to authenticate the user, the Panopticon Streams will only know if the user is authorized or not. To authenticate the user, Panopticon Streams can be configured to use a REST service to extract the user identity with an access token retrieved from the OAuth 2.0 provider. In addition to the standard OAuth 2.0 configurations, the server includes properties (i.e., authentication.oauth2.*) that are specifically used to extract the user details.

- authentication.type=OAUTH2
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication.oauth2.client.id</td>
<td>The ID of the OAuth 2.0 client.</td>
</tr>
<tr>
<td>authentication.oauth2.client.secret</td>
<td>The secret used by the OAuth 2.0 client.</td>
</tr>
<tr>
<td>authentication.oauth2.identity.attribute.username</td>
<td>The attribute that will be extracted from the identity response and used as the username.</td>
</tr>
<tr>
<td>authentication.oauth2.identity.url</td>
<td>The URL to the REST service that provides details about the authenticated user.</td>
</tr>
<tr>
<td>authentication.oauth2.login.callback.url</td>
<td>The callback URL. The URL should be the same as one of the specified callback URLs used by the client. The URL should refer to the Panopticon Streams.</td>
</tr>
<tr>
<td>authentication.oauth2.login.response.type</td>
<td>The response type. The only response type that is currently supported is <strong>CODE</strong>. The value can also be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.scope</td>
<td>The requested scope. The field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.state</td>
<td>The requested state. The field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.url</td>
<td>The URL to the OAuth 2.0 login resource.</td>
</tr>
<tr>
<td>authentication.oauth2.logout.url</td>
<td>The URL to the OAuth 2.0 logout resource. This field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.redirect</td>
<td>Redirects the user back to the Panopticon Streams URL. This is mainly used with a proxy. In which case, the Panopticon Streams does not know the endpoint which the user is going towards to, and therefore cannot redirect the user back to the Overview page. This can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.token.method</td>
<td>The method on how the token should be retrieved. Supported values are <strong>QUERY</strong>, <strong>BODY</strong>, and <strong>HEADER</strong>.</td>
</tr>
<tr>
<td>authentication.oauth2.token.url</td>
<td>The URL to the OAuth 2.0 token resource.</td>
</tr>
</tbody>
</table>
Custom authentication filters can be applied to the server and the application when the default authentication settings are not sufficient. This type of authentication is referred to as **Filter authentication**. When the Panopticon Streams is configured to use filter authentication, it means that the incoming requests have already been authenticated and authorized before reaching the server. Follow the steps below to configure filter authentication:

1. Open the `streams.properties` file in the `AppData` folder (`c:\streamsserverdata`).
2. Enable `authentication.type=FILTER` in `streams.properties`.
3. Apply the following URL pattern to your own filter: `/*`
4. Save the changes and restart the Tomcat.

**CREATING A CUSTOM FILTER**

The custom filter will be a basic authentication filter which will authenticate the user with hardcoded values. The Principal forwarded by the filter will be used to authenticate the user.

The filter will require the following dependencies:
- Javax Servlet
- Tomcat embed core

**Steps:**

1. Create a HTTP request wrapper.
The class will contain the following:

- the original incoming HTTP request
- the Principal which contains both the credentials and the roles for the authenticated user.

The HTTP wrapper will be forwarded to the Panopticon Streams instead of the original incoming HTTP request.

```java
import org.apache.catalina.realm.GenericPrincipal;
import org.apache.catalina.users.MemoryUser;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletRequestWrapper;
import java.security.Principal;
public class FilterRequestWrapper extends HttpServletRequestWrapper {
    private final GenericPrincipal principal;
    public FilterRequestWrapper(final HttpServletRequest request, final GenericPrincipal principal) {
        super(request);
        this.principal = principal;
    }
    @Override
    public Principal getUserPrincipal() {
        return principal;
    }
    @Override
    public boolean isUserInRole(final String role) {
        if (principal != null) {
            return principal.hasRole(role);
        }
        return super.isUserInRole(role);
    }
}
```

2. Create a custom filter. The filter will create a new Principal which includes both the credentials and the groups/roles for the user.

In this example, the class `GenericPrincipal` contains username, password, and groups. The Panopticon Streams is only able to extract the groups from `GenericPrincipal` class or the `MemoryUser` class. Both the Principal and the original HTTP request will be wrapped in an instance of `FilterRequestWrapper`. The wrapper will then be forwarded towards the Panopticon Streams.

```java
import org.apache.catalina.realm.GenericPrincipal;
import org.apache.catalina.users.MemoryUser;
import javax.servlet.*;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.security.Principal;
import java.util.Arrays;
import java.util.List;
public class ExampleFilter implements Filter{
```
@Override
public void init(FilterConfig filterConfig) throws ServletException {}

@Override
public void doFilter(final ServletRequest servletRequest, final ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {
    if (!(servletRequest instanceof HttpServletRequest || !(servletRequest instanceof HttpServletResponse))) {
        return;
    }

    final HttpServletRequest request = (HttpServletRequest) servletRequest;
    final HttpServletResponse response = (HttpServletResponse) servletResponse;
    final String username = "username";
    final String password = "password";
    final List<String> groups = Arrays.asList("Group1", "Group2");
    final GenericPrincipal principal = new GenericPrincipal(username, password, groups);
    filterChain.doFilter(new FilterRequestWrapper(request, principal), response);
}

@Override
public void destroy() {}

3. When these classes have been created, you can compile them and package them in a jar file.

4. Copy the jar file to the WEB-INF/lib folder in the panopticon war file (or the extracted folder).

5. Enable the filter by adding the following code to the web.xml file in panopticon WEB-INF folder:

```xml
<filter>
  <filter-name>ExampleFilter</filter-name>
  <filter-class>com.datawatch.server.filter.ExampleFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>ExampleFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```
Header

It is possible to use a web-facing Panopticon Streams behind a proxy server that will handle the authentication of users. The proxy server forwards the name of the user and roles to the Panopticon Streams as HTTP headers for every request.

For requests where headers are blank or missing, they are treated like anonymous requests while requests where the user HTTP header are valid are treated like authenticated requests with that specific username.

Requests from the proxy server are fully trusted and checks are no longer performed at the Panopticon Streams with regards to the validity of the username. The authorization on workbooks and administration will work as usual.

To activate the Header authentication, add or update the following properties in the streams.properties file:

```properties
authentication.type=HEADER
authentication.header.role.delimiter=,
authentication.header.roles={roles header, ie. X-Roles}
authentication.header.username={userid header, ie. X-User}
```
Adding Administrators

You can assign users that were defined in Tomcat User Base Realm, LDAP, or Active Directory, as administrators on the System tab.

Steps:

1. Log on to the Panopticon Streams using your authentication credentials (i.e., designer/designer).

2. Click the System tab.

3. Enter a user (e.g., John) in the Users box, this enables the Add button.

4. Click Add.
Since you logged on as **designer**, it now has no administrator privileges. A message box is displayed.

Click **OK**.

The other tabs will display errors.

**NOTES**

Logging on as **designer** will only display the **Engine** tab.

---

**John** now has administrator rights while the rest of the users in `tomcat-users.xml` will be forbidden to use Panopticon Streams.

5. Log out as **designer** and log on as **John**. It is now added in the list.

Also, the `Administrators.txt` file is added in the AppData folder (i.e., `c:\streamsserverdata`). **John** is added in the list.
6. You can delete an administrator by clicking \[\times\]. A confirmation message displays.

```
Are you sure you want to remove "john" from the list of administrators?

YES  NO
```

7. Click Yes.

**NOTES**

If for example, the only administrator in the list is deleted (e.g., John), then all of the users defined in Tomcat will have administrator privileges again.

---

**Adding Administrator Group**

You can assign roles or groups that were defined in Tomcat User Base Realm, LDAP, or Active Directory, as administrator groups on the System tab.

**NOTES**

It is recommended to have administrator users already added on the System tab before adding administrator groups.

For example:

```
ADMINISTRATORS

Users

Add

John
mary
```

**Steps:**

1. Log on as an administrator (e.g., John) and click System tab.
2. Enter a group name in the Groups box (e.g., Administrators), this enables the Add button.
3. Click **Add**. The group is added in the list.

   ![Administrators](image)

   Also, the `AdministratorGroups.txt` file is added in the **AppData** folder (i.e., c:\streamsserverdata).

4. You can delete an administrator group by clicking **X**. A confirmation message displays.

5. Click **Yes**.

### Reloading Configurations

The **System** tab supports reloading of applications, data sources, administrators, and parameters from the file system.
Click **Reload Configuration**. This will stop and restart applications and reload data sources along with the administrators and parameters.

The Panopticon Streams is supplied with a command line utility PCLI.jar.

Export Data Sources

THE PCLI provides functionality to export data sources from one or all workbooks in a directory. The exported data sources can be uploaded and used directly by the Panopticon Streams.

PARAMETERS

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>DESCRIPTION</th>
<th>REQUIRED</th>
</tr>
</thead>
<tbody>
<tr>
<td>-w, --workbook</td>
<td>The name of the workbook.</td>
<td>Yes (or -wd)</td>
</tr>
<tr>
<td>-od, --output-directory</td>
<td>The output directory where the data source will be exported to.</td>
<td>No</td>
</tr>
<tr>
<td>-wd, --workbook-directory</td>
<td>The directory of the workbooks folder.</td>
<td>Yes (or -w)</td>
</tr>
<tr>
<td>-dd, --data-directory</td>
<td>The directory of the data folder.</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --license-file</td>
<td>The path of the license file.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Example 1: Export data sources from a workbook

```
java -jar pcli.jar exportdatasource
-w "C:/Panopticon/Workbooks/VizGuide.exw"
-l "C:/Panopticon/DatawatchLicense.xml"
-dd "C:/Panopticon/Data"
-od "C:/Panopticon/CEP/Datasources"
```
Example 2: Export data sources from all workbooks example

```
java -jar pclij.jar exportdatasource
   -wd "C:/Panopticon/Workbooks"
   -l "C:/Panopticon/DatawatchLicense.xml"
   -dd "C:/Panopticon/Datawatch/Data"
   -od "C:/Panopticon/CEP/Datasources"
```
[6] Using Altair Panopticon Streams

Connecting to or Disconnecting from the CEP Engine

Upon logging on to the Panopticon Streams, the default web page displays the following properties:

![Image of Panopticon Streams Engine tab]

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Displays whether the Panopticon Streams is connected to or disconnected from the CEP Engine (Kafka).</td>
</tr>
<tr>
<td>Type</td>
<td>The CEP Engine that the Panopticon Streams engine will work with (KAFKA).</td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>The URL to the ZooKeeper servers. Default is localhost:2181.</td>
</tr>
<tr>
<td>Kafka</td>
<td>The URL of all the Kafka servers. Default is localhost:9092.</td>
</tr>
<tr>
<td>Schema Registry</td>
<td>The URL to the Schema Registry. Default is <a href="http://localhost:8081">http://localhost:8081</a>.</td>
</tr>
</tbody>
</table>
CONNECTING TO THE CEP ENGINE:

There are two ways to use Panopticon Streams with Kafka. You can either connect to an external or an internal Kafka setup.

To connect to an external setup:

NOTES

Ensure the ZooKeeper, Kafka, and Schema Registry batch files are already running before connecting to the engine.

Click CONNECT to connect to the external Kafka.

To connect to an internal setup:

1. Open your streams.properties file and set attribute cep.kafka.path to the value of the Streams.zip file content path (e.g., if the content is placed in C:\Streams you set cep.kafka.path=C:/Streams).
2. Restart the Panopticon Streams server.
3. Click CONNECT to connect to the internal Kafka.

NOTES

- The CEP engine must be running to be able to start applications and generate the input and output topics.
- If you are running Panopticon Streams with the internal Kafka setup, make sure to only use the forward slash '/' in the path value of cep.kafka.path

DISCONNECTING FROM THE CEP ENGINE:

Click DISCONNECT. Consequently, the applications cannot be started and the input and output topics will not be generated.
[7] Managing Applications

Applications can be uploaded, defined, and started on the Applications tab.

Figure 7-1. Applications tab

Uploading Applications

Steps:

1. On the Applications tab, click Upload Application. The Upload Application dialog displays.
2. To upload an application, you can either:
   - drag it from your desktop and drop in the dialog, or
   - click **Choose Application** and select one in the *Open* dialog that displays.

3. You can opt to rename the application.

**NOTES**

The application name must start with a letter (a to Z) or underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores.

4. To replace an existing application, check the *Replace existing application* box.
5. Click **Upload**.
   
   You will be notified when the application has been uploaded.
The application is added and displayed on the **Applications** tab.

![Upload Application](image)

NOTES

A ![Alert](image) icon displays before the application name. This means the required data source is not available. Refer to [Uploading Data Sources](#) for more information.

When the data source is available, the icon changes to ![Check](image).
SORTING THE LIST OF APPLICATIONS

By default, the list of applications is sorted by Name in an ascending order.

You can modify the sorting of the list by clicking the or button of the Name, Last Modified, or Status Updated columns. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.
SEARCHING FOR APPLICATIONS

To search for a particular application, enter it in the Filter Applications box.

You can also enter one of more characters into the Filter Applications box and the suggested list of applications that matched the entries will be displayed.
Creating a New Application

Steps:

1. On the Applications tab, click New Application. The Application page displays with the title (e.g., UntitledApplication_0).

The Application page displays with the following sections. Initially, the Operator Settings section is displayed.

Clicking \( \text{[Metrics panel]} \) displays the Metrics panel:
Clicking  displays the Data Preview panel. Note that you need to run the application to preview the data.

<table>
<thead>
<tr>
<th>SECTION/PANEL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Name</td>
<td>Name of the application. Click the  button to go back to the Applications listing page.</td>
</tr>
<tr>
<td>Error Validation</td>
<td>After saving the changes in the application, this allows error validation. If there are definition issues (red node) or if there is no traffic on the topic (yellow node), you can click  to help fix the errors. If there are no issues,  is no longer displayed in the Application page.</td>
</tr>
<tr>
<td>Add Operator</td>
<td>The available operators that can be added in the application.</td>
</tr>
<tr>
<td>Delete Operator</td>
<td>Delete the selected operator.</td>
</tr>
<tr>
<td>Save</td>
<td>Save the changes made in the Application page.</td>
</tr>
<tr>
<td>Save Options and Properties</td>
<td>Allow saving of changes made in the application or saving another copy. It also displays the application properties as well as adding new ones.</td>
</tr>
<tr>
<td>Expand Graph</td>
<td>Expand the Graph panel.</td>
</tr>
<tr>
<td>Stop Application</td>
<td>Stop the execution of the application.</td>
</tr>
<tr>
<td>Run Application</td>
<td>Run or execute an application.</td>
</tr>
</tbody>
</table>
### Reference Guide 17.4.0

**Graph**
Display the nodes and edges of the application model. It allows single node selection.

**View Metrics**
Display the throughput of the selected nodes (total and current message rates) in the Metrics panel. When the application is running, the metrics data are refreshed.

**Data Preview**
Display the retrieved query or table in the Data Preview panel.

**Maximize**
Expand the Operator Settings, Metrics, or Data Preview panel.

**View Operator Settings**
When an operator has been added or a node is selected in the application graph, the corresponding operator settings are displayed in the Operator Settings panel to allow editing. When the application is running, the operator settings are displayed but are not editable.

**Schema**
Display the preview of the data.

---

Refer to the sections below to proceed in creating an application.

---

### Adding an Operator

Operators can be added in any order. The sequence or direction of the graph will be based on the inputs or outputs that will connect the nodes.

When adding an operator, its node will be displayed in the **Graph** panel.

![Adding an Operator](image)

---

**NOTES**

- The edges (inbound and/or outbound) will depend on the operator.
- For best practice, start by adding Input operators (i.e., Input, External Input, or Metronome) and end with the Output operator.
Also, the corresponding Operator Properties and Schema are displayed in the Application page.

**ADDING AN INPUT OPERATOR**

Used to define the input data for the application model.

Steps:

1. In the Application page, click + and select Input in the Context menu that displays.

The Input node icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.
This operator serves as the initial source of the data in the application. The right (outbound) edge allows you to connect to other operators.

2. In the Operator Settings panel, define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>The ID of the input operator.</td>
</tr>
<tr>
<td>Topic</td>
<td>The stream of records or input you will be subscribed to. Check the Use Node ID box to use the value entered in the Input ID. Otherwise, uncheck the box and enter a new Topic ID. When adding Topic IDs, ensure they: must be unique across an application must be specified must start with a letter (a to Z) or an underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores</td>
</tr>
<tr>
<td>Input Type</td>
<td>Select the input type: STREAM, TABLE, or GLOBAL_TABLE.</td>
</tr>
<tr>
<td>Data Source</td>
<td>Select the data source. <strong>NOTES:</strong> It is recommended to upload the data source first so they will be available for selection. Selecting a non-streaming data source displays the Refresh Period (ms) property. Enter the refresh period for the data.</td>
</tr>
</tbody>
</table>
This value determines when to periodically reload the data (from the beginning).

The preview of the data (OUTPUT) is displayed in the Schema panel.

| Key Columns | The key column. Proceed to step 3. |

**NOTE:** *Input, Topic, Input Type, and Data Source* properties are required.

3. In the *Key Columns* section, click + to add a key column from the data source schema. Repeat to add more.

You can also delete a key column in the list by checking its box and clicking -.

4. **Save** the changes.

**Example**

```
<inpu>
  <id>Input</id>
  <topic>Input</topic>
  <dataProducer>
    <id>StocksStatic</id>
    <refreshPeriod>1000</refreshPeriod>
    <keyColumns>
      <field>Id</field>
    </keyColumns>
  </dataProducer>
  <inputType>TABLE</inputType>
</inpu>
```

**ADDING AN AGGREGATION OPERATOR**

The aggregation operator aggregates the data based on a grouping key and a set of aggregated fields.

**Steps:**

1. In the *Application* page, click + and select **Aggregation** in the Context menu that displays.

```
  aggregation1
```

The **Aggregation** node icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
This operator has left (inbound) and right (outbound) edges that allow connection to other operators in the application.

5. In the *Operator Settings* panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggregation</td>
<td>The ID of the aggregation operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>Automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the <em>Inputs</em> drop-down list. The preview of the data (INPUT) is displayed in the <em>Schema</em> panel.</td>
</tr>
<tr>
<td>Time Window</td>
<td>Select the time window for the aggregation operator (in milliseconds).</td>
</tr>
<tr>
<td>Enable Cache</td>
<td>Specifies whether to start or stop caching tables. When caching is stopped, it gets every event input into the table to produce an output event. This is necessary for delta/prev aggregates.</td>
</tr>
<tr>
<td>Group By</td>
<td>The name/IDs of the columns that the data will be grouped by. (Proceed to step 3.)</td>
</tr>
<tr>
<td>Fields List</td>
<td>A set of aggregated fields (with actions and expressions). (Proceed to step 5.)</td>
</tr>
</tbody>
</table>
6. In the *Group By* section, click +. A column is added in the list. Click the drop-down list to select another column.

7. Select a column that will be used to group the data.

The INPUT and OUTPUT schema are displayed.
Repeat steps 3 and 4 to add more.

You can also delete a column in the Group By list by checking its box and clicking .

8. In the Field List section, click . A new field entry displays.

9. Enter the Field Name and the Expression that will be evaluated for each incoming record.
   Example:
   Field Name: Count
   Expression: count()

10. Select the Add action.
    Repeat steps 6 and 7 to add more aggregated fields.
You can also:

- check the topmost box to select all of the fields

- change the order of the fields by checking a field’s box and clicking either the ▲ or ▼ button

- delete a field entry in the Field List by checking its box and clicking —

The OUTPUT schema is updated based on the added aggregations.

11. **Save** the changes.
Example

```xml
<aggregation>
  <id>Aggregation</id>
  <fields>
    <field>
      <id>ColumnId</id>
      <action>ADD</action>
      <expression>Sum(Mcap_USD)</expression>
    </field>
  </fields>
  <groupBy>
    <field>Industry</field>
  </groupBy>
</aggregation>
```

Supported Aggregation Functions

This section lists the aggregation functions that are only supported in aggregation operator expressions.

**NOTES**

- All of the supported [calculation functions](#) and [operators](#) can be used in aggregations.
- Panopticon Streams also supports nullability where:
  - a field may or may not allow null/empty/missing/NA values
  - functions or operators may or may not allow null arguments (e.g., you can’t divide seven by null)

<table>
<thead>
<tr>
<th>AGGREGATION FUNCTION</th>
<th>DESCRIPTION</th>
<th>NULLABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>avg(X)</td>
<td>The average or mean of all non-null values in X</td>
<td>✔</td>
</tr>
<tr>
<td>collect(X)</td>
<td>Takes as its argument a column of any type and creates a nested table of the input type out of the rows selected</td>
<td>✔</td>
</tr>
<tr>
<td>count(X)</td>
<td>The count of the number of records in the selection</td>
<td></td>
</tr>
<tr>
<td>delta(X)</td>
<td>last(expression) - first(expression)</td>
<td>✔</td>
</tr>
<tr>
<td>first(X)</td>
<td>The first value in X including the nulls</td>
<td>✔</td>
</tr>
<tr>
<td>firstNonNull(X)</td>
<td>The first value in X that is not null</td>
<td>✔</td>
</tr>
<tr>
<td>last(X)</td>
<td>The last value in X including the nulls</td>
<td>✔</td>
</tr>
<tr>
<td>lastNonNull(X)</td>
<td>The last value in X that is not null</td>
<td>✔</td>
</tr>
</tbody>
</table>
max(X) | The maximum value in X
---|---
min(X) | The minimum value in X
prev(X) | Collects expression into an array of size 2, then returns the element at index 0
samples(X) | The number of non-null values in X
sdevp(X) | The Population Standard Deviation of X
sdevs(X) | The Sample Standard Deviation of X
sum(X) | Calculates the total or sum of the non-null values in X
varp(X) | Returns the population variance of all non-null numeric values specified by the expression, evaluated in the context of the given scope
vars(X) | Returns the sample variance of all non-null numeric values specified by the expression, evaluated in the context of the given scope
wavg(X) | Weighted Average

**NOTES**

The following aggregates work with a time window (can subtract): count, samples, sum, sdev, var, and avg

**Building the Expression**

- When building the expression, take note that the column name is case sensitive
- A validation error displays with a suggestion to help build the expression

Examples:

```
Add ▼ Samples  samples(MCAP_USD)
```

Col 8: Unable to find column MCAP_USD, did you mean Mcap_USD

Click the link (e.g., [Mcap(USD)]) to apply the correct entry.

```
Add ▼ Samples  samples()
```

col 8: Something missing? Got <nothing> expected or '?'

Complete the expression as necessary.

**Converting Timestamp to/from Integer**

Allows you to convert Timestamp values to/from Integer which include the following examples:
from posix to timestamp
from posixmillis to timestamp
from timestamp to posix
from timestamp to posixmillis

The conversion uses the expression: `to([typename],[expression],[format])`

Examples:
- `to(int, timefieldname, 'POSIX')`
- `to(time, intfieldname, 'POSIX')`

**ADDING A BRANCH OPERATOR**

The branch operator will split a stream into one or more branches. The path for a stream is determined by a configured predicate within the branch operator.

The predicate expression will be evaluated for each incoming record. A record will be routed to the first branch with a matching predicate.

**Steps:**

1. In the Application page, click ![Branch icon] and select Branch in the Context menu that displays.

   ![Branch icon]

   The Branch icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.
The left (inbound) edge allows you to connect to an input data or operator. The right (outbound) edges allow you to add more streams.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch</td>
<td>The ID of the branch operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>Automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. The preview of the data (INPUT/OUTPUT) is displayed in the Schema panel.</td>
</tr>
<tr>
<td>Predicate</td>
<td>A list of predicates. Each predicate contains an expression that will be evaluated for each record.</td>
</tr>
</tbody>
</table>

3. To add more predicates, click +. A new predicate entry displays. Enter at least two expressions.
You can also:

- check the topmost box to select all of the fields
- delete a field entry in the Field List by checking its box and clicking

4. **Save** the changes.

**Example**

```xml
<branch>
  <id>Branch</id>
  <predicates>
    <!-- One_Day_Change < 0 -->
    <predicate>One_Day_Change < 0</predicate>
    <!-- One_Day_Change >= 0 -->
    <predicate>One_Day_Change >= 0</predicate>
  </predicates>
</branch>
```

**Example 2**

```xml
<streams>
  <stream>
    <source>Input</source>
    <sink>
      <operator>Branch</operator>
    </sink>
  </stream>
  <stream>
    <source>Branch</source>
    <port>1</port>
    <sink>
      <operator>Output1</operator>
    </sink>
  </stream>
  <stream>
    <source>Branch</source>
    <port>2</port>
    <sink>
      <operator>Output2</operator>
    </sink>
  </stream>
</streams>
```
ADDING A CALCULATION OPERATOR

The calculation operation will calculate a field and add the result as an additional field. Usually, input fields pass through an operation, but calculations can also be set to replace existing fields or simply remove them.

Steps:

1. In the Application page, click + and select Calculation in the Context menu that displays.

   ![Calculation node icon and Graph panel](image)

   The Calculation node icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.

   ![Operator Settings panel and Schema panel](image)

   This operator has left (inbound) and right (outbound) edges that allow connection to other operators in the application.

2. In the Operator Settings panel, define or select the following required properties:
### PROPERTY | DESCRIPTION
--- | ---
Calculation | The ID of the calculation operator.
Inputs | Automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. The preview of the data (INPUT and OUTPUT) are displayed in the Schema panel.
Fields List | Set of fields (with actions and expressions). Enter at least one calculated field. Proceed to step 3.

3. Enter the **Field Name** and the **Expression** that will be evaluated for each incoming record.

4. Select any of the following actions: **Add**, **Replace**, or **Remove**.

5. Click **+** to add a new field entry and repeat steps 3 and 4.

The OUTPUT schema is updated based on the added calculations.

You can also:
- check the topmost box to select all of the fields
- change the order of the fields by checking a field’s box and clicking either the **↑** or **↓** button
- delete a field entry in the **Field List** by checking its box and clicking **−**
6. Save the changes.

SUPPORTED OPERATORS AND CALCULATION FUNCTIONS

This section lists the supported operators and calculation functions in Panopticon Streams.

NOTES

Panopticon Streams supports nullability where:

- a field may or may not allow null/empty/missing/NA values
- functions or operators may or may not allow null arguments (e.g., you can’t divide seven by null)

Supported Operators

These are typically the operators that are used to create aggregation operator and calculation operator expressions.

<table>
<thead>
<tr>
<th>OPERATOR</th>
<th>DESCRIPTION</th>
<th>NULLABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Logical NOT</td>
<td></td>
</tr>
<tr>
<td>!=</td>
<td>Not equal to</td>
<td></td>
</tr>
<tr>
<td>%</td>
<td>Calculates the modulo (division remainder) of two numbers</td>
<td></td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical AND</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>Multiply</td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>Add</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>Subtract</td>
<td></td>
</tr>
<tr>
<td>/</td>
<td>Divide</td>
<td></td>
</tr>
<tr>
<td>&lt;</td>
<td>Lesser than</td>
<td></td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equals</td>
<td></td>
</tr>
<tr>
<td>==</td>
<td>Equal to</td>
<td></td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td></td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equals</td>
<td></td>
</tr>
<tr>
<td>?:</td>
<td>Ternary if</td>
<td></td>
</tr>
<tr>
<td>^</td>
<td>Raises to the power</td>
<td></td>
</tr>
<tr>
<td>like</td>
<td>Tests if a string matches a pattern</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Supported Calculation Functions

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>DESCRIPTION</th>
<th>NULLABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs</td>
<td>Absolute Value</td>
<td></td>
</tr>
<tr>
<td>array</td>
<td>Creates an array</td>
<td>✓</td>
</tr>
<tr>
<td>atan</td>
<td>ArcTangent</td>
<td></td>
</tr>
<tr>
<td>ceil</td>
<td>Ceiling</td>
<td></td>
</tr>
<tr>
<td>clock</td>
<td>Processor time</td>
<td></td>
</tr>
<tr>
<td>concat</td>
<td>Concatenates two strings together</td>
<td></td>
</tr>
<tr>
<td>cos</td>
<td>Cosine</td>
<td></td>
</tr>
<tr>
<td>cosh</td>
<td>Cosine Hyperbolic</td>
<td></td>
</tr>
<tr>
<td>cot</td>
<td>Cotangent</td>
<td></td>
</tr>
<tr>
<td>dateDiff</td>
<td>Calculates the difference in whole units between two time values. Valid datepart values are YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, MILLISECONDS, MICROSECONDS, and NANOSECONDS.</td>
<td></td>
</tr>
<tr>
<td>dateDiff2</td>
<td>The total amount of elapsed time between two time values expressed in a given unit. Valid datepart values are YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, MILLISECONDS, MICROSECONDS, and NANOSECONDS.</td>
<td></td>
</tr>
<tr>
<td>datePart</td>
<td>Returns the datepart information about a specified date/time as a character string</td>
<td></td>
</tr>
<tr>
<td>dateTrunc</td>
<td>Truncates date/time to a specified precision</td>
<td></td>
</tr>
<tr>
<td>exp</td>
<td>Exponential</td>
<td></td>
</tr>
<tr>
<td>find</td>
<td>Starting point of a string within a target string</td>
<td></td>
</tr>
<tr>
<td>floor</td>
<td>Floor</td>
<td></td>
</tr>
<tr>
<td>get</td>
<td>Returns the array element at the specified index</td>
<td>✓</td>
</tr>
<tr>
<td>if</td>
<td>Conditional Statement</td>
<td>✓</td>
</tr>
<tr>
<td>ifNull</td>
<td>If the first expression is not NULL, it returns the first expression. Otherwise, it returns the second expression</td>
<td>✓</td>
</tr>
<tr>
<td>index</td>
<td>Given an array of N values, returns an array of integers of the same length N, with the values 1 through N ordered to reflect the natural order of the argument values</td>
<td></td>
</tr>
<tr>
<td>intpow</td>
<td>Raises a value to an integral power</td>
<td></td>
</tr>
<tr>
<td>invert</td>
<td>Since the index function returns an inverse permutation, you can apply the invert function which will turn it into a forward permutation (or rank)</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>left</td>
<td>Extracts a substring from a string, starting from the left-most character</td>
<td></td>
</tr>
<tr>
<td>len</td>
<td>Returns the length of the specified string</td>
<td></td>
</tr>
<tr>
<td>In</td>
<td>Natural Log</td>
<td></td>
</tr>
<tr>
<td>log</td>
<td>10 Based Log</td>
<td></td>
</tr>
<tr>
<td>logn</td>
<td>Returns the Log Based N of Input</td>
<td></td>
</tr>
<tr>
<td>lower</td>
<td>To Lower Case</td>
<td></td>
</tr>
<tr>
<td>max</td>
<td>Maximum of two input values</td>
<td></td>
</tr>
<tr>
<td>mid</td>
<td>Substring from the middle of an input string</td>
<td></td>
</tr>
<tr>
<td>min</td>
<td>Minimum of two input values</td>
<td></td>
</tr>
<tr>
<td>pow</td>
<td>Raises a value to a power</td>
<td></td>
</tr>
<tr>
<td>proper</td>
<td>To Title Case</td>
<td></td>
</tr>
<tr>
<td>random</td>
<td>Random Number</td>
<td></td>
</tr>
<tr>
<td>replaceAll</td>
<td>Returns a string after it replaces each substring that matches the given regular expression and replacement string</td>
<td></td>
</tr>
<tr>
<td>replaceFirst</td>
<td>Replace the first instance of the pattern_to_replace with the replacement_text. For example: replaceFirst(input_text, pattern_to_replace, replacement_text) replaceFirst(&quot;ABA&quot;, &quot;A&quot;, &quot;X&quot;) = &quot;XBA&quot; Note: Only input_text may be null. Special cases: • If input_text is null, the result is null. • If pattern_to_replace is empty, it's considered to occur at every position in the input_text (including before the first and after the last character).</td>
<td></td>
</tr>
<tr>
<td>right</td>
<td>Right most characters of the input string</td>
<td></td>
</tr>
<tr>
<td>rnd</td>
<td>Returns a random number</td>
<td></td>
</tr>
<tr>
<td>round</td>
<td>Returns a number rounded to a given number of digits</td>
<td></td>
</tr>
<tr>
<td>set</td>
<td>Sets the array element at the specified index, returns the resulting array</td>
<td></td>
</tr>
<tr>
<td>sign</td>
<td>Returns 1 if Positive, 0 if Zero, -1 if Negative</td>
<td></td>
</tr>
<tr>
<td>sin</td>
<td>Sine</td>
<td></td>
</tr>
<tr>
<td>sinh</td>
<td>Sine Hyperbolic</td>
<td></td>
</tr>
<tr>
<td>sort</td>
<td>Given an array V of N values and an integer array P of the same size containing a permutation of the values 1,2, ..., N, returns an array S of length N with the values from V permuted according to P.</td>
<td></td>
</tr>
</tbody>
</table>
The array P can be a forward or reverse permutation which is dependent on what the index function returns. Ultimately, the sort(V, index(V)) should return V sorted with the smallest element first.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sqr</td>
<td>Square</td>
</tr>
<tr>
<td>sqrt</td>
<td>Square Root</td>
</tr>
<tr>
<td>tan</td>
<td>Tangent</td>
</tr>
<tr>
<td>trim</td>
<td>Strips leading and following spaces</td>
</tr>
<tr>
<td>trunc</td>
<td>Discards the fractional part of a number</td>
</tr>
<tr>
<td>upper</td>
<td>To Upper Case</td>
</tr>
</tbody>
</table>

**Example**

```xml
<calculation>
  <id>Calculation</id>
  <fields>
    <field>
      <id>SquareRoot_ColumnA</id>
      <action>ADD</action>
      <expression>SquareRoot(ColumnA)</expression>
    </field>
  </fields>
</calculation>
```

**ADDDING A CONFLATE OPERATOR**

The conflate operation is used to lower the frequency of updates. The conflate will retain the last records seen on the input and push them to the output stream on a fixed interval. For example, if the input is producing a high frequency data throughput, instead of processing all of these updates, a configured conflate will only push through a small set of records on a fixed interval.

**Steps:**

1. In the *Application* page, click and select **Conflate** in the Context menu that displays.

The **Conflate** node icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
This operator has left (inbound) and right (outbound) edges that allow connection to other operators in the application.

2. In the **Operator Settings** panel, define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conflate</td>
<td>The ID of the conflate operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>Automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the <em>Inputs</em> drop-down list. The preview of the data (INPUT and OUTPUT) are displayed in the <em>Schema</em> panel.</td>
</tr>
<tr>
<td>Interval</td>
<td>The interval of which the data should be published to the output stream (in milliseconds).</td>
</tr>
<tr>
<td>Timestamp</td>
<td>The timestamp.</td>
</tr>
<tr>
<td>Keep Records</td>
<td>Check to retain or not remove flushed elements. This means the entire set of records will be flushed at each interval.</td>
</tr>
</tbody>
</table>

**NOTE:** *Conflate, Inputs, and Interval* are required properties.

3. **Save** the changes.
Example

```xml
<conflate>
  <id>Conflate</id>
  <interval>10000</interval>
</conflate>
```

**ADDING AN EXTERNAL INPUT**

Sources data directly from a Kafka topic.

Steps:

1. In the *Application* page, click ![add icon] and select **External_input** in the Context menu that displays.

   ![external_input1]

   The **External Input** icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
This operator serves as the initial source of the data in the application. The right (outbound) edge allows you to connect to other operators.

2. In the Operator Settings panel, define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Externa Input</td>
<td>The ID of the external input operator.</td>
</tr>
<tr>
<td>Input Type</td>
<td>Select the input type: <strong>STREAM</strong>, <strong>TABLE</strong>, or <strong>GLOBAL_TABLE</strong>.</td>
</tr>
<tr>
<td>Topic</td>
<td>The stream of records or input you will be subscribed to.</td>
</tr>
<tr>
<td>From Beginning</td>
<td>Check to retrieve full history (from beginning to the latest) of the topic. If un-checked, only the latest messages after the application has started will be retrieved.</td>
</tr>
</tbody>
</table>

**NOTE:** *External_input*, *Input Type*, and *Topic* properties are required.

3. Select the Parser Plugin:
   - XML
• JSON

Parser Plugin | json
Record Path
Name | Json Path | Type | Date Format
Name | STRING

• Text

If Text has been selected, confirm the Column Delimiter and Text Qualifier, and if the first row of the message includes column headings.

Parser Plugin | text
Column Delimiter
Text Qualifier
First Row Headings
Name | Index | Type | Date Format
First Row Headings
Name | STRING

4. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>XPath/Json Path/Index</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be: BOOLEAN, DATE, DATETIME, DOUBLE, FLOAT, INT, LONG, STRING, TIME.</td>
</tr>
<tr>
<td>Date Format</td>
<td>Date/Time format when the data type is DATE, DATETIME, or TIME.</td>
</tr>
</tbody>
</table>

5. You can also click the following icons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Add a new field entry.</td>
</tr>
</tbody>
</table>
Check a box of a field entry and click to delete.

Fetch the schema of the output topic. This populates the list of columns, with the data type found from inspecting the first 'n' rows of the file.

6. Save the changes.

**ADDING A FILTER OPERATOR**

Used to filter a data source based on a predicate.

Steps:

1. In the Application page, click and select Filter in the Context menu that displays.

The Filter icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.
This operator has left (inbound) and right (outbound) edges that allow connection to other operators in the application.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter</td>
<td>The ID of the filter operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>Automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. The preview of the data (INPUT/OUTPUT) is displayed in the Schema panel.</td>
</tr>
<tr>
<td>Predicate</td>
<td>Determines whether the input record will be included or excluded. The records that will not match the predicate will be filtered out and will not be part of the output result. Example:</td>
</tr>
</tbody>
</table>

```
<filter>
  <id>Filter</id>
  <!-- One_Day_Change >= 0 -->
  <predicate>One_Day_Change >= 0</predicate>
</filter>
```

3. Save the changes.

Example

**ADDING A JOIN OPERATOR**

Used to join data sources using common keys.

Steps:

1. In the Application page, click and select Join in the Context menu that displays.

The Join node icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.
The left (inbound) edges allow you to select the input sources or operators that will be joined. The right (outbound) edge allows you to connect to other operators.

2. In the **Operator Settings** panel, define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Join</td>
<td>The ID of the join operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The left input automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. Then select the right input. The preview of the data (LEFT, RIGHT, and OUTPUT) are displayed in the <strong>Schema</strong> panel.</td>
</tr>
<tr>
<td>Time Window</td>
<td>The time window for the join operation (in milliseconds).</td>
</tr>
<tr>
<td>Join Type</td>
<td>The type of the join: <strong>INNER</strong>, <strong>LEFT</strong>, or <strong>OUTER</strong></td>
</tr>
<tr>
<td>Left Field</td>
<td>The columns from the left source that will be used to join with.</td>
</tr>
<tr>
<td>Right Field</td>
<td>The columns from the right source that will be used to join with.</td>
</tr>
</tbody>
</table>

**NOTE:** **Join**, **Inputs**, **Join Type**, and **Left Field** with **Right Field** properties are required.

3. In the **Fields List** section, click †. The key columns of the left and right sources are automatically displayed.
Repeat step 3 to add more columns.

You can also:

- check the topmost box to select all of the fields
- delete a field entry in the Field List by checking its box and clicking

The LEFT, RIGHT, and OUTPUT schema are displayed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ticker</td>
<td>string (key, null)</td>
</tr>
<tr>
<td>Date</td>
<td>datetime (null)</td>
</tr>
<tr>
<td>Adj_Close</td>
<td>double (null)</td>
</tr>
<tr>
<td>Period_Change_pc</td>
<td>double (null)</td>
</tr>
<tr>
<td>Volume</td>
<td>double (null)</td>
</tr>
<tr>
<td>Turnover</td>
<td>double (null)</td>
</tr>
<tr>
<td>SPS500_Change</td>
<td>double (null)</td>
</tr>
<tr>
<td>Relative_Change</td>
<td>double (null)</td>
</tr>
<tr>
<td>Holding</td>
<td>double (null)</td>
</tr>
</tbody>
</table>
4. **Save** the changes.

**Example**

```xml
<join>
  <id>Join</id>
  <joinType>INNER</joinType>
  <left>
    <field>Ticker</field>
  </left>
  <right>
    <field>Ticker</field>
  </right>
  <timeWindow>1000</timeWindow>
</join>
```
Fixing Broken Joins

Changes in the input data sources may cause issues in the Join operator of an application like broken joins and output schema.

For example, if the original data source contains Brand and Country columns:
And if eventually the column Country is deleted in the data source, opening the application again will display:

Click **Manually Update Output Schema** to fix this error. Note that Country is no longer in the list of the Output schema.
Click to apply the changes.

On the other hand, if new columns are added in the data source (e.g., if the Country column is added in the data source again), opening the application will display:

Click **Manually Update Output Schema**. Note that Country is added in the Output schema.
Click [button] to apply the changes.

**ADDING A METRONOME INPUT OPERATOR**

Similar with a synthetic input, this operator acts as a single timestamp field schema generator.

**Steps:**

1. In the *Application* page, click [button] and select *Metronome* in the Context menu that displays.

   ![metronome1]

   The *Metronome* node icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
The right (outbound) edge allows you to connect to the other operators.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metronome</td>
<td>The ID of the metronome operator.</td>
</tr>
<tr>
<td>Topic</td>
<td>The stream of records or input you will be subscribed to. Check the Use Node ID box to use the value entered in the Input ID. Otherwise, uncheck the box and enter a new Topic ID. When adding Topic IDs, ensure they: • must be unique across an application • must be specified • must start with a letter (a to Z) or an underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores</td>
</tr>
<tr>
<td>Interval</td>
<td>The interval of which the data should be published to the output stream.</td>
</tr>
<tr>
<td>Name Field Id</td>
<td>The ID of the name field.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the ID.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>The name of the new column that will include the timestamp.</td>
</tr>
</tbody>
</table>
The preview of the data (OUTPUT) is displayed in the Schema panel.

3. **Save** the changes.

Example

```xml
<metronome>
  <id>Metronome</id>
  <topic>Metronome</topic>
  <dataProducer>
    <bufferSize>0</bufferSize>
    <id>Metronome</id>
    <keyColumns/>
    <refreshPeriod>100</refreshPeriod>
  </dataProducer>
  <inputType>STREAM</inputType>
  <interval>100</interval>
  <name>Metronome</name>
  <nameFieldId>ID</nameFieldId>
  <timestampFieldId>Timestamp</timestampFieldId>
</metronome>
```

**ADDING A PYTHON TRANSFORM OPERATOR**

A Python script can be executed as a data transformation step in the data pipeline.

Steps:

1. In the Application page, click ![add icon] and select **Python Transform** in the Context menu that displays.

   ![Python Transform icon]

   The **Python Transform** node ![node icon] icon displays in the Graph panel, as well as the properties to be defined in the **Operator Settings** panel, and the preview of the data in the Schema panel.
The right (outbound) edge allows you to connect to the other operators.

2. In the **Operator Settings** panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Python Transform</td>
<td>The ID of the Python Transform operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The stream of records or input you will be subscribed to.</td>
</tr>
</tbody>
</table>

Reference Guide 17.4.0 77
<table>
<thead>
<tr>
<th><strong>Interval</strong></th>
<th>The interval of which the data should be published to the output stream (in milliseconds).</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Keep Records</strong></td>
<td>Check to retain or not remove flushed elements. This means the entire set of records will be flushed at each interval.</td>
</tr>
<tr>
<td><strong>Host</strong></td>
<td>Host of the Python Pyro instance.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Port of the Python Pyro instance.</td>
</tr>
<tr>
<td><strong>HMAC Key</strong></td>
<td>The HMAC key that will be used to connect to the Python Pyro instance.</td>
</tr>
<tr>
<td><strong>Data Object Name</strong></td>
<td>The data structure (array of dictionaries) that Panopticon Designer will produce, and then will be utilized by the Python Script.</td>
</tr>
</tbody>
</table>
| **Serialization Type** | The serialization type: Serpent or Pickle  
• simple serialization library based on `ast.literal_eval`  
• faster serialization but less secure |

3. Enter the required Python script to execute on the active Pyro instance.

4. You can also click the following icons:

<table>
<thead>
<tr>
<th><strong>BUTTON</strong></th>
<th><strong>DESCRIPTION</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>⌛</td>
<td>Fetch the schema of the output topic. This populates the list of columns, with the data type found from inspecting the first ‘n’ rows of the file.</td>
</tr>
<tr>
<td>+</td>
<td>Add a new field entry.</td>
</tr>
<tr>
<td>−</td>
<td>Check a box of a field entry and click − to delete.</td>
</tr>
</tbody>
</table>

5. **Save** the changes.
Example

<operators>
  <transform>
    <id>python_transform1</id>
    <transformPlugin>Python</transformPlugin>
    <transformPluginSettings/>
    <interval>1000</interval>
    <columns>
      <type>STRING</type>
    </columns>
    <maxRowsCount>0</maxRowsCount>
  </transform>
  <input>
    <id>input1</id>
    <topic>input1</topic>
    <globalTopic>UntitledApplication_0.input1</globalTopic>
    <dataProducer>
      <id>StreamSimulator_StocksStatic</id>
      <keyColumns>
        <field>Region</field>
      </keyColumns>
    </dataProducer>
    <inputType>STREAM</inputType>
  </input>
  <output>
    <id>output1</id>
    <topic>output1</topic>
    <globalTopic>UntitledApplication_0.output1</globalTopic>
    <dataConsumer>TextOutput</dataConsumer>
  </output>
</operators>

<streams>
  <stream>
    <source>python_transform1</source>
    <sink>
      <operator>output1</operator>
    </sink>
  </stream>
  <stream>
    <source>input1</source>
    <sink>
      <operator>python_transform1</operator>
    </sink>
  </stream>
</streams>

**ADDING A RANK OPERATOR**

Assign a rank number to records in the same group.
Steps:

1. In the **Application** page, click ![+] and select **Rank** in the Context menu that displays.

   ![Graph panel showing Rank node icon and operator settings]

   The **Rank** node icon displays in the **Graph** panel, as well as the properties to be defined in the **Operator Settings** panel, and the preview of the data in the **Schema** panel.

   ![Operator Settings panel with Rank and Inputs properties]

   The right (outbound) edge allows you to connect to the other operators.

2. In the **Operator Settings** panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank</td>
<td>The ID of the rank operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The stream of records or input you will be subscribed to.</td>
</tr>
</tbody>
</table>
**Interval** | How often the collected data should be sorted, ranked, and output (in milliseconds)
---|---
**Rank Field** | The name of the rank number field in the output.
**Rank Percentile** | The name of the percentile field in the output. This is the rank number divided by the number of records in the group.
**Reference** | The input field to sort records on when ranking them.
**Sort Order** | The order to sort the records: ASCENDING (the lowest value gets rank one) or DESCENDING (the highest value gets rank one).
**Cumulative Sum** | The cumulative sum based on the currently applied sort order for each Reference value. You can opt to specify a new value in the Into field. This column will be added in the Output schema.
**Group By** | The name/IDs of the fields that the data will be grouped by. Records are ranked within each group. (Proceed to step 3.)

3. In the **Group By** section, click +. A column is added in the list. Click the drop-down list to select the column that will be used to group the data.
The INPUT and OUTPUT schema are displayed.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td>string (key, null)</td>
</tr>
<tr>
<td>Country</td>
<td>string (null)</td>
</tr>
<tr>
<td>Exchange</td>
<td>string (null)</td>
</tr>
<tr>
<td>Name</td>
<td>string (null)</td>
</tr>
<tr>
<td>Forex</td>
<td>string (null)</td>
</tr>
<tr>
<td>Symbol</td>
<td>string (null)</td>
</tr>
<tr>
<td>ISIN</td>
<td>string (null)</td>
</tr>
<tr>
<td>SEDOL</td>
<td>string (null)</td>
</tr>
<tr>
<td>Close_local</td>
<td>double (null)</td>
</tr>
<tr>
<td>Mcap_local</td>
<td>double (null)</td>
</tr>
<tr>
<td>Mcap_USD</td>
<td>double (null)</td>
</tr>
<tr>
<td>Industry</td>
<td>string (null)</td>
</tr>
<tr>
<td>Supersector</td>
<td>string (null)</td>
</tr>
<tr>
<td>One_Day_Close</td>
<td>double (null)</td>
</tr>
<tr>
<td>One_Week_Close</td>
<td>double (null)</td>
</tr>
<tr>
<td>Two_Week_Close</td>
<td>double (null)</td>
</tr>
<tr>
<td>One_Month_Close</td>
<td>double (null)</td>
</tr>
</tbody>
</table>

Repeat step 3 to add more.

Two columns are added in the Output schema: <Rank Field> and the <Rank Percentile>. For example:
If you set a name for the *Cumulative Sum*, it will also be added in the Output schema.

You can also delete a column in the *Group By* list by checking its box and clicking "-".

4. **Save** the changes.
Example

```xml
<rank>
  <id>rank1</id>
  <interval>1000</interval>
  <groupBy>
    <field>Region</field>
    <field>Country</field>
  </groupBy>
  <rankField>rank</rankField>
  <rankPercentileField>rank_percentile</rankPercentileField>
  <reference>Mcap_USD</reference>
  <sortOrder>ASCENDING</sortOrder>
  <cumSumSourceField>One_Week_Change</cumSumSourceField>
</rank>
```

**ADDING A REKEY OPERATOR**

Takes a stream data and changes its key. The new key can be any subset of fields from the stream.

**NOTES**

This operator can never be applied to a table since tables require keys to be unique and therefore, you need to specify how multiple records with the same key should be aggregated.

**Steps:**

1. In the *Application* page, click + and select **Rekey** in the Context menu that displays.

   ![Rekey node icon](image)

   The **Rekey** node icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
The right (outbound) edge allows you to connect to the other operators.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rekey</td>
<td>The ID of the rekey operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The stream of records or input you will be subscribed to.</td>
</tr>
<tr>
<td>Key</td>
<td>The key column. Proceed to step 3.</td>
</tr>
</tbody>
</table>

**NOTE:** Rekey, Inputs, and Key properties are required.

3. In the Key section, click 🗓️ to select the new key column in the drop-down list box from the data source schema. Repeat to add more.

You can also delete a key column in the list by checking its box and clicking ✗.

The preview of the data (OUTPUT) is displayed in the Schema panel.
4. **Save** the changes.

Example

```
<rekey>
  <id>rekey1</id>
  <key>
    <field>Ticker</field>
    <field>Volume</field>
  </key>
</rekey>
```

**ADDING A SCATTER OPERATOR**

Given a record with array fields (must have the same length), the scatter operator will emit one record for each position in the array(s). This operator is similar with unpivot but on array positions instead of columns.

If the input record has an integer array field A of length N and text (non-array) field T, the operator will output N records with integer (non-array) field A and text (non-array) field T. For example, they will have values: \( \{ A[0], T \}, \{ A[1], T \}, \ldots, \{ A[N-1], T \} \) (assuming zero-based indexing).

If the input has no array fields, the scatter operator is a no-op and will pass records through unchanged.

**Steps:**

1. In the *Application* page, click \( + \) and select **Scatter** in the Context menu that displays.

The **Scatter** icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
The right (outbound) edge allows you to connect to the other operators.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scatter</td>
<td>The ID of the scatter operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The stream of records or input you will be subscribed to.</td>
</tr>
</tbody>
</table>

**NOTE:** *Scatter* and *Inputs* properties are required.

The preview of the data (OUTPUT) is displayed in the Schema panel.
3. **Save** the changes.

**Example**

```xml
<scatter>
   <id>scatter1</id>
</scatter>
</operators>
<streams>
   <stream>
      <source>input1</source>
      <sink>
         <operator>scatter1</operator>
      </sink>
   </stream>
</streams>
```

**ADDING A TABLE TO STREAM OPERATOR**

Aggregating on delta as a Table causes a change log, producing a single record. The Table to Stream operator morphs the single record back into Stream.

**Steps:**

1. In the *Application* page, click `➕` and select **To_stream** in the Context menu that displays.

   ![to_stream1](image)

   The **To_stream** node icon displays in the *Graph* panel, as well as the properties to be defined in the *Operator Settings* panel, and the preview of the data in the *Schema* panel.
The right (outbound) edge allows you to connect to the other operators.

2. In the **Operator Settings** panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>To_stream</td>
<td>The ID of the Table to Stream operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The left input automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the <strong>Inputs</strong> drop-down list. Ideally, this should be an aggregation operator. The preview of the data (LEFT, RIGHT, and OUTPUT) are displayed in the <strong>Schema</strong> panel.</td>
</tr>
</tbody>
</table>

The preview of the data (OUTPUT) is displayed in the **Schema** panel.
NOTES

The data types of the aggregated columns are still unknown. The new data type will be applied once the application is saved.

For example:

3. **Save** the changes.

Example

```
<tostream>
  <id>to_stream1</id>
</tostream>
```
ADDING A UNION OPERATOR

Used to perform a union of two streams. Both streams would need the same schema. Otherwise, the output would be the combination of both, with missing values returned as Null.

Steps:

1. In the Application page, click ![Add Operator](image) and select Union in the Context menu that displays.

The Union node ![Union Node](image) icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.

The left (inbound) edges allow you to select the input streams. The right (outbound) edge allows you to connect to the other operators.

2. In the Operator Settings panel, define or select the following required properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Union</td>
<td>The ID the union operator.</td>
</tr>
<tr>
<td>Inputs</td>
<td>The left input stream automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. Then select the right input stream.</td>
</tr>
</tbody>
</table>
3. Save the changes.

Example

```
<union>
  <id>Union</id>
</union>
```

ADDITION AN OUTPUT OPERATOR

An output produces and publishes streams towards a Kafka topic or a data consumer.

Steps:

1. In the Application page, click and select Output in the Context menu that displays.

The Output node icon displays in the Graph panel, as well as the properties to be defined in the Operator Settings panel, and the preview of the data in the Schema panel.
The left (inbound) edge allows you to select the input source or operator.

2. In the Operator Settings panel, define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>The ID of the output operator.</td>
</tr>
</tbody>
</table>
| Topic     | The stream of records or output you will be subscribed to. Check the Use Node ID box to use the value entered in the Output ID. Otherwise, uncheck the box and enter a new Topic ID. When adding Topic IDs, ensure they:
  • must be unique across an application
  • must be specified
  • must start with a letter (a to Z) or an underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores |
| Inputs    | The left input stream automatically connects to the currently-selected operator. You can select another ID of the operator that will be the source of the data in the Inputs drop-down list. The preview of the data (INPUT) is displayed in the Schema panel. |
| Data Consumer | Select the Data Consumer where the output will be produced or published. Currently, the following data consumers are supported:  
  • Text |
3. **Save** the changes.

Example 1

```xml
<output>
    <id>Output</id>
    <topic>Output</topic>
</output>
```

Example 2

```xml
<output>
    <id>TextExampleOutput</id>
    <topic>Output</topic>
    <dataConsumer>TextOutput</dataConsumer>
</output>
```

### Adding Application-specific Properties

Panopticon Streams properties can be viewed and configured in *streams.properties*. However, some of these Server-wide properties can be overridden by adding and customizing them in an application.

**Steps:**

1. In the *Application* page, click 🔄 then select **Properties**.
   The *Application Properties* dialog displays.

2. To add a property, click 📦.
   A new row for *Key* and *Value* entry displays.

**NOTES**

Currently, the application properties are used to assign specific retention time (in milliseconds) for topic(s).
NOTES

The Keys and Values must not be empty. Also, keys must be unique within the application property list.

3. Enter the Key. This is the application property to be configured.
4. Enter the corresponding Value of the key.
   You can also opt to delete an application property entry by checking its box and clicking.
5. Click Close.

Example

<properties>
   <!-- Keep tables alive one day -->
   <entry>
      <key>table.retention.ms</key>
      <value>86400000</value>
   </entry>

   <!-- Keep input and output streams for 1 second -->
   <entry>
      <key>input.retention.ms</key>
      <value>1000</value>
   </entry>
   <entry>
      <key>output.retention.ms</key>
      <value>1000</value>
   </entry>

   <!-- Custom retention time for InputStream topic -->
   <entry>
      <key>TimeSeries.retention.ms</key>
      <value>1111</value>
   </entry>
</properties>

Refer to RetentionTimeExample in the Example Applications section for more information.
Saving an Application

Saved applications (.app) are available in the <DatawatchVDDAppdata>\CEP\Applications folder (i.e., c:\streamsserverdata\CEP\Applications).

Steps:

1. In the Application page, you can either click:
   - the Save icon
   - the  icon. The context menu displays with three saving options:
     - Save
       Click to save the changes made in the application.
     - Save as Copy
       Click to make a duplicate of the application. The original name is appended with _Copy.
       To change the Application Name, click on it to make it editable, then enter a new one and click .

   ![NOTES]
   The Name or ID must start with a letter (a to Z) or underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores.

   - Revert to Saved
     Click to revert to the previously-saved application settings.
NOTES

- After saving, you can start the execution of the application. You can do this either in the *Application* page or on the *Applications* tab.
- When saving an application, the color will indicate the status of the nodes:
  - Black – no issue
  - Yellow – no traffic on the topic
  - Red – there are definition issues. Refer to *Validating and Fixing Application Issues* for more information.

Editing an Application

NOTES

Applications that are started or running cannot be edited.

Steps:
1. On the *Applications* tab, click an application link to modify. The *Application* page displays.
2. To change the *Application Name*, click on it to make it editable, then enter a new one and click ✔.

NOTES

The *Name* or ID must start with a letter (a to Z) or underscore. Also, it can only contain letters (a to Z), numbers (0 to 9), and underscores.
3. You can also modify or add the following:
   - operators
   - properties

4. Save the changes.

To go back to the Applications tab, click ← beside the application name.

NOTES

If there are changes that were not saved, a confirmation message displays asking if you will exit the Application page without saving. Click Cancel and then save.

Validating and Fixing Application Issues

Panopticon Streams provides an error validation to help fix application issues.

Steps:

1. Click . The list of Issues is displayed with the Source or operator with an error.
2. Click the link of the operator with an issue.
   Some possible issues:
   - for the input nodes, the data source is not available
   - the application model parts are still not complete, or has invalid values
   - for all nodes except inputs, there are faulty input definition or missing input connection
   - for all nodes except outputs, there are faulty output definition or missing output connection
3. Apply the necessary changes and save.
Starting an Application

NOTES

• Before starting an application, ensure:
  • the CEP engine has been started
  • the prerequisite data sources are uploaded on the Data Sources tab
  • the application model is defined correctly

• If the application is empty, the icon is disabled. Refer to Creating a New Application for more information.

You can start an application either on the Applications tab or on the Application page.

STARTING AN APPLICATION ON THE APPLICATIONS TAB

Figure 7-3. Running applications on the Applications tab
Steps:

1. To execute an application, ensure the green icon displays before the Name. This means the necessary data sources are already uploaded.
   - However, if orange is displayed, click it to display the Application Data Sources message. For example:

     ![Application Data Sources](image)

     StockMarketSimulator - Datasources
     Below is the list of datasources that currently use 'StockMarketSimulator':
     - StockMarketSimulatorSymbols

     Refer to [Uploading Data Sources](#) for more information.
   - Other possible error is:

     ![Application could not be started](image)

     Application could not be started
     Failed to build application.

     Refer to [Editing an Application](#) and [Validating and Fixing Application Issues](#) for more information.

2. Click  . The icon changes to red and the timestamp is displayed under the Status Updated column.
   - This also generates the stream topics and the data producer.
STARTING AN APPLICATION ON THE APPLICATION PAGE

Steps:

1. On the Applications tab, click an application link to open and display it on the Application page.
2. Click ⏯️ to run the application.

This also enables the ⏸️ icon and generates the stream topics and the data producer.

3. You can also perform the following:

- click on a node in the Graph panel and ⚙️ to display its Operator Settings as well as the preview of the data (OUTPUT) in the Schema panel
• click 📊 to display the node’s Metrics as well as the preview of the data (OUTPUT) in the Schema panel
• select all the nodes and click ![to display all of their throughput values (total and current message rates)](image) in the *Metrics* panel
- select an input or output node and click 🔄 to display the data preview
Stopping an Application

Stopping the execution of an application can either be done on the Applications tab or the Application page.

**STOPPING AN APPLICATION ON THE APPLICATIONS TAB**

Click ![Stop button](image) to stop the execution of the application. The icon is changed back to ![Running button](image).
STOPPING AN APPLICATION ON THE APPLICATION PAGE

Click \( \bigtriangledown \) to stop the execution of the application. The \( \bigtriangledown \) is enabled.

Viewing and Managing Application Topic Usages

On the Applications tab, you can view the input and output topic usages of an application.

Steps:

1. Click the \( Ti \) icon of an application.
   The <Application Name> Topics dialog displays.
   If the application has been executed, the #Messages column will display the number of retrieved messages, while the Messages/sec column will display the number of retrieved messages per second.
   If the application is not yet executed, both the #Messages and Messages/sec columns will display 0 values.
2. You can then opt to:
3. Click **Close**.

### CLEARING THE DATA IN AN APPLICATION TOPIC

You can recycle an application by:

1. **stopping** the application
2. deleting data in the topics
3. **restarting** the application

Follow the steps below to clear the data in an application topic.

**Steps:**

1. You can either:
   - Click **DATA** to delete the data in a topic, or
   - Click **Clear All Data** to delete the data in all of the topics in an application
2. Click **Close**.

### CLEARING THE SCHEMA IN AN APPLICATION TOPIC

Schema registry can be cleared in any application topic.

**Steps:**

1. You can either:
   - click **SCHEMA** to delete the schema in a topic, or
   - click **Clear Schemas** to delete the schema in all of the topics in an application
2. Click **Close**.

### SORTING APPLICATION TOPICS

By default, the list of topics is sorted by *Name* in an ascending order.
You can modify the sorting of the list by clicking the ▼ or ▲ button of the Name, Type, #Messages or Messages/sec columns. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.

Other Applications Operations

On the Applications tab, you can also perform the following:

- View Application History and Republish
- View the application data sources usage
- Download a copy
- Delete an application
- View other pages of the applications list

VIEWING APPLICATION HISTORY AND REPUBLISHING

This section describes the following operations:

- Viewing the change history of applications
- Republishing an archived application to the recent version of Panopticon Streams
- Renaming an archived application

Steps:

1. On the Applications tab, click the History button of an application.

   The <Application Name> - History dialog displays:
You can sort the archival list either through the *Date Archived* or *Archived By* columns by clicking on the ▼ or ▲ button.

Also, move to the other pages of the list by clicking on a page or clicking the « or » button.

2. Click on an archived application in the list.

Then click **Republish**. A notification message displays.

3. Click **Yes**. A notification message displays.

4. Click **OK**.
5. You may also opt to rename an archived application by entering a new one in the New Name box and follow steps 2 to 4 to republish it.

VIEWING THE APPLICATION DATA SOURCES USAGE

On the Applications tab, you can view the data sources that are currently used by an application.

Steps:

1. Click the icon of an application.
   The list of data sources that is currently used by the application displays.
2. Click Close.

DOWNLOADING AN APPLICATION

Click the of an application to download and save a copy.

DELETING AN APPLICATION

Steps:

1. Click the icon of an application.
   A confirmation message displays.
2. Click Yes to delete.

MOVING TO OTHER APPLICATIONS TAB PAGES

Go to the other Applications tab pages by clicking:
- any link of a page number
- . This displays the previous page
- . This displays the next page
[8] Managing Data Sources

On the **Data Sources** tab, you can perform the following:

- **Upload data sources**
- **Create** new data source
- **Modify** data source properties
- **View application usages**
- **Download** a copy
- **Delete** a data source
- Search for a data source

### Uploading Data Sources

**Steps:**

1. On the **Data Sources** tab, click **Upload Data Source**. The **Upload Data Source** dialog displays.
2. To upload an application data source, you can either:
   • drag it from your desktop and drop in the dialog, or
   • click **Choose Data Source** and select one in the *Open* dialog that displays.

3. You can opt to rename the application data source.

4. To replace an existing application data source, check the *Replace existing data source* box.

5. Click **Upload**.
   You will be notified when the application data source has been uploaded.
Creating a Data Source

Panopticon Streams supports creation of data sources that can be used as inputs or outputs in the application model.

Steps:

1. On the Data Sources tab, click New Data Source.

The Data Source tab displays with the following sections:

<table>
<thead>
<tr>
<th>SECTION/PANEL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>Name of the data source. Click the button to go back to the Data Sources listing page.</td>
</tr>
</tbody>
</table>
2. Enter the Name of the data source. This should be unique and should only contain letters (a to Z), numbers (0 to 9), and underscores.

3. Click ✅ or press Enter to apply the name.

4. Select any of the following:
   - output connectors
     - Email
     - InfluxDB
     - JDBC Databases
     - Apache Kafka
     - Kx kdb+
     - Rest
     - Text
   - Input data sources
     - ActiveMQ
     - AMPS
     - InfluxDB
     - JDBC Databases
     - JSON
     - Apache Kafka
     - Kx kdb+
     - Kx kdb+ tick
     - MQTT
     - MS Excel
     - OneTick Cloud
     - RabbitMQ
     - Solace
     - Stream Simulator
     - Text
     - WebSocket
     - XML
The tab page changes depending on the selected connector. Refer to the sections below for more information.

CREATING EMAIL OUTPUT CONNECTOR

Steps:

1. On the Data Source tab, select Output > Email in the Connector drop-down list.

   ![EmailDataSource](image)

2. Define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Email host address.</td>
</tr>
<tr>
<td>Port</td>
<td>Email host port.</td>
</tr>
<tr>
<td>Mail Security Mode</td>
<td>Select the email security mode: NONE, SSL, or TLS</td>
</tr>
<tr>
<td>Sender Email Address</td>
<td>Email address of the sender.</td>
</tr>
<tr>
<td>Sender Password</td>
<td>Password of the sender.</td>
</tr>
<tr>
<td>To Email Address</td>
<td>Email address of the recipient.</td>
</tr>
<tr>
<td>CC Email Address</td>
<td>Email address of the CC recipient.</td>
</tr>
<tr>
<td>BCC Email Address</td>
<td>Email address of the BCC recipient.</td>
</tr>
<tr>
<td>Subject</td>
<td>Subject of the email.</td>
</tr>
<tr>
<td>Text</td>
<td>Content of the email.</td>
</tr>
</tbody>
</table>

3. Click ![Submit](image). The new data source is added in the Data Sources list.
CREATING INFLUXDB OUTPUT CONNECTOR

Allows periodical dumping of data from a Kafka topic into a time series database such as InfluxDB.

Steps:

1. On the Data Source tab, select Output > Influx DB in the Connector drop-down list.

2. Define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>URL of the InfluxDB.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that will be communicate over the HTTP(S).</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to InfluxDB.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that will be used to connect to InfluxDB.</td>
</tr>
<tr>
<td>Measurement</td>
<td>The table name that can be used as measurement.</td>
</tr>
</tbody>
</table>

3. Click +. A new column entry displays. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Target</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
</tbody>
</table>

4. Repeat step 3 to add more sources.
5. You can also click any of the following buttons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Check" /></td>
<td>Check the topmost <img src="image" alt="Check" /> to select all data source column entries.</td>
</tr>
<tr>
<td><img src="image" alt="Check" /></td>
<td>Check the <img src="image" alt="Check" /> of a data source entry or check the topmost <img src="image" alt="Check" /> to select all column entries and click <img src="image" alt="Delete" /> to delete.</td>
</tr>
<tr>
<td><img src="image" alt="Select" /></td>
<td>Allows you to select an output topic in the drop-down list.</td>
</tr>
</tbody>
</table>

Click **OK**. The schema of the selected output topic is displayed.

Tests if the connection to the output connector is successful. If successful, a confirmation message displays.

Click **OK**. Otherwise, an error prompt displays.
CREATING JDBC DATABASE OUTPUT CONNECTOR

Allows periodical dumping of records from a Kafka topic into a JDBC database.

Steps:

1. On the Data Source tab, select Output > JDBC in the Connector drop-down list.

2. You can either select:
   - JNDI Name
     Enter the JNDI resource name to be used, then the Username and Password.
   - URL
     Enter the URL specific to the database’s JDBC driver, the Driver Class Name specific to the driver, and the Username and Password.

3. Select the appropriate SQL Dialect in the drop-down list to be able to generate the correct SQL for the required data repository.
   You can select any of the following SQL dialects: AnsiSQL, Access/Excel, MySQL, Oracle, SQL Server, Sybase IQ/ASA, Sybase ASE, Netezza, Vertica, SQLite, HadoopHive, KxQ, DB2, PostgreSQL, Impala, Redshift, Informix, Teradata, dBase, SparkSQL.

4. Enter the source Table (can be parameterized).

5. In the Sources section, click +. A new source entry is added in the list.

6. Click Close. The new data source is added in the Data Sources list.
6. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Target</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
</tbody>
</table>

7. Repeat steps 5 and 6 to add more sources.
8. You can also click any of the following buttons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Check the topmost to select all data source column entries.</td>
</tr>
<tr>
<td></td>
<td>Check the of a data source entry or check the topmost to select all column entries and click to delete.</td>
</tr>
<tr>
<td></td>
<td>Allows you to select an output topic in the drop-down list.</td>
</tr>
</tbody>
</table>

Select an output topic

CANCEL OK
Click **OK**. The schema of the selected output topic is displayed.

Tests if the connection to the output connector is successful. If successful, a confirmation message displays.

Click **OK**. Otherwise, an error prompt displays.

Click **Close** and fix the connection error.

9. Click **Add**. The new data source is added in the *Data Sources* list.
CREATING APACHE KAFKA OUTPUT CONNECTOR

Allows publishing of events to an external Kafka JSON or Avro topic. For Avro, ensure to point towards the schema registry used by the external Kafka cluster.

Steps:

1. On the Data Source tab, select Output > Kafka in the Connector drop-down list.

![KafkaOutput](image)

2. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZooKeeper Host</td>
<td>Where the Kafka Server is located.</td>
</tr>
<tr>
<td>ZooKeeper Port</td>
<td>The port number of ZooKeeper being used by the Kafka server. Default is 2181.</td>
</tr>
<tr>
<td>Schema Registry Host</td>
<td>Where the Schema Registry is located. This can be in a different location from the Kafka cluster.</td>
</tr>
<tr>
<td>Schema Registry Port</td>
<td>The port number of the schema registry which provides the serving layer for the metadata. Default is 8081.</td>
</tr>
</tbody>
</table>

3. Enter the Topic name.
4. Select the Message Composer: Avro or JSON
5. In the Sources section, click +. A new source entry is added in the list.
6. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Target</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
</tbody>
</table>

7. Repeat steps 5 and 6 to add more sources.

8. You can also click any of the following buttons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Check the topmost button to select all data source column entries.</td>
</tr>
<tr>
<td></td>
<td>Check the of a data source entry or check the topmost button to select all column entries and click to delete.</td>
</tr>
<tr>
<td></td>
<td>Allows you to select an output topic in the drop-down list.</td>
</tr>
</tbody>
</table>
Click **OK**. The schema of the selected output topic is displayed.

Tests if the connection to the output connector is successful. If successful, a confirmation message displays.

Click **OK**.
Otherwise, an error prompt displays.

Click **Close** and fix the connection error.

9. Click **. The new data source is added in the *Data Sources* list.
CREATING KX KDB+ OUTPUT CONNECTOR

Allows periodical dumping of records from a Kafka topic into a Kx kdb+ connector.

Steps:

1. On the Data Source tab, select Output > Kdb+ in the Connector drop-down list.

2. Define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Kx kdb+ host address.</td>
</tr>
<tr>
<td>Port</td>
<td>Kx kdb+ host port. Default is 5001.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to Kx kdb+.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that will be used to connect to Kx kdb+.</td>
</tr>
<tr>
<td>Table</td>
<td>The source Table (can be parameterized).</td>
</tr>
</tbody>
</table>

3. In the Sources section, click . A new source entry is added in the list.
4. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Target</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
</tbody>
</table>

5. Repeat steps 3 and 4 to add more sources.

6. You can also click any of the following buttons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Check the topmost to select all data source column entries.</td>
</tr>
<tr>
<td></td>
<td>Check the of a data source entry or check the topmost to select all column entries and click to delete.</td>
</tr>
<tr>
<td></td>
<td>Allows you to select an output topic in the drop-down list.</td>
</tr>
</tbody>
</table>

Click **OK**. The schema of the selected output topic is displayed.

Tests if the connection to the output connector is successful. If successful, a confirmation message displays.
7. Click . The new data source is added in the Data Sources list.

**CREATING A REST OUTPUT CONNECTOR**

Outputs an event to a REST API. This output connector can also be used as an alerting system.

**Steps:**

1. On the Data Source tab, select Output > Rest in the Connector drop-down list.
2. Define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>URL of the REST API.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to REST API.</td>
</tr>
<tr>
<td>Password</td>
<td>The user name that will be used to connect to REST API.</td>
</tr>
<tr>
<td>HTTP Method</td>
<td>Select the HTTP Method to map any of the following operations to HTTP requests</td>
</tr>
<tr>
<td></td>
<td>• GET – retrieve information</td>
</tr>
<tr>
<td></td>
<td>• POST – create or update an entity</td>
</tr>
<tr>
<td></td>
<td>• PUT – replace an existing entity</td>
</tr>
<tr>
<td></td>
<td>• DELETE – remove a request</td>
</tr>
<tr>
<td>PROPERTY</td>
<td>DESCRIPTION</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Content Type</td>
<td>The required Content Type. Default is <strong>application/x-www-form-urlencoded</strong></td>
</tr>
<tr>
<td>Timeout</td>
<td>The length of time to wait for the server response (10 to 300). Default is <strong>10</strong>.</td>
</tr>
<tr>
<td>Request Body</td>
<td>The Request Body for HTTP POST.</td>
</tr>
</tbody>
</table>

3. Click ![image](image). The new data source is added in the *Data Sources* list.

**CREATING TEXT OUTPUT CONNECTOR**

Allows retrieval and processing of delimited Text files (such as CSV, TSV, etc.). The files produced can be consumed by the Text connector.

**Steps:**

1. On the *Data Source* tab, select **Output > Text** in the *Connector* drop-down list.

2. Define or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Folder Path</td>
<td>The path where the Text output will be placed.</td>
</tr>
<tr>
<td>File Name Prefix</td>
<td>The prefix for the file name. This can be parameterized with field names. Consequently, each event can generate a new file in the given folder. For example, if the Text output connector is attached as the consumer to StockStatic, you can use &quot;{Region}&quot; inside the <em>File Name Prefix</em>, causing it to create three files for Asia Pacific, Europe, and North America. Note that partitioning file names with current date in &quot;yyyyMMdd&quot; format is still done automatically and can't be controlled, at the moment. For the StockStatic example, if it was executed today, it would have created three files like Asia Pacific_20181219.tsv.</td>
</tr>
<tr>
<td>Timestamp Column</td>
<td>The name of the new column that will include the timestamp. Default is <strong>Timestamp</strong>.</td>
</tr>
</tbody>
</table>

3. In the *Sources* section, click ![image](image). A new source entry is added in the list.
4. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Target</td>
<td>The column name of the target schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
<tr>
<td>Output Date Format</td>
<td>The output Date/Time format when the data type is <strong>Time</strong>.</td>
</tr>
</tbody>
</table>

5. Repeat steps 3 and 4 to add more sources.

6. You can also click any of the following buttons:

<table>
<thead>
<tr>
<th>BUTTON</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Check the topmost ☐ to select all data source column entries.</td>
</tr>
<tr>
<td></td>
<td>Check the ☐ of a data source entry or check the topmost ☐ to select all column entries and click — to delete.</td>
</tr>
<tr>
<td>☐</td>
<td>Allows you to select an output topic in the drop-down list.</td>
</tr>
</tbody>
</table>
Click **OK**. The schema of the selected output topic is displayed.

Tests if the connection to the output connector is successful. If successful, a confirmation message displays.

Click **OK**. Otherwise, an error prompt displays.

Click **Close** and fix the connection error.

7. Click **. The new data source is added in the *Data Sources* list.

**CREATING ACTIVEMQ INPUT DATA SOURCE**

Allows connection to Apache’s ActiveMQ message bus on a real-time streaming basis. Specifically, the connector allows Panopticon Streams to subscribe to XML, JSON or FIX based messages that are published on topics. The data format itself is arbitrary, and consequently, the connection includes the message definition.
Steps:

1. In the *New Data Source* page, select **Input > Active MQ** in the *Connector* drop-down list.

![ActiveMQInput](image)

2. Enter the following information:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker</td>
<td>The location of the message broker.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to the ActiveMQ service.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to connect to the ActiveMQ service.</td>
</tr>
<tr>
<td>Topic</td>
<td>The topic or queue physical name.</td>
</tr>
</tbody>
</table>

3. Check/uncheck the **Use durable subscription** box.

**NOTES**

When connecting to a message bus, it is recommended to **disable** durable messaging. When it is enabled, this puts a heavier load to the Server, and slows down the start and stop of subscriptions.

4. Select the **Message Type**:
   - FIX
5. Select either the dot (.) or comma (,) as the Decimal Separator.

### NOTES

Prepend 'default:' for the elements falling under default namespace.

6. Click 

   to add columns to the MQ connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Fix Tag/JsonPath/Text Column Index/XPath</td>
<td>The Fix Tag/JsonPath/Text Column Index/XPath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>
NOTES

To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them.

For example: yyyy-MM-dd HH:mm:ss.fffffff

To delete a column, check its [ ] or all the column entries, check the topmost [ ] , then click [ ] .

7. Instead of manually adding columns, you can opt to click [ ] to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

8. If the Type is selected as Text, it will be listed in the Id Column drop-down list box and can be used to select a key column to manage data updates and inserts.

   **Note:** Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

Furthermore, a streaming time series window can be generated by creating a compound key with the Id Column, plus a separately specified Time ID column. This Time ID column can be from the source dataset, or alternatively automatically generated.

If the Time Id column is selected, then a scrolling time window can be specified.

   Time Id Column [Automatic Time Id] 
   Time Id Column Name Automatic_TIMESTAMP_Column

For **Automatic Time Id**, define the Time Id Column Name.

As new data arrives from the subscription, new time slices will automatically be added, and old ones will be deleted.

If a new ID is received, a new row is added to the in-memory data set representing the ActiveMQ topic subscription. While if an existing ID is received, an existing row is updated.

9. Click [ ]. The new data source is added in the Data Sources list.

CREATING AMPS INPUT DATA SOURCE

The AMPS connector allows connection to AMPS message bus on a real-time streaming basis. The connector allows Panopticon Streams to subscribe to the Native FIX and XML message support. The data format itself is arbitrary, and in turn the connection includes the message definition.

Steps:

1. In the New Data Source page, select Input > AMPS in the Connector drop-down list.
2. Enter the following information:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>AMPS host address.</td>
</tr>
<tr>
<td>Port</td>
<td>AMPS host port. Default is 9004.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to the AMPS service.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to connect to the AMPS service.</td>
</tr>
<tr>
<td>Topic</td>
<td>The topic or queue physical name.</td>
</tr>
<tr>
<td>Filter</td>
<td>The filter expression.</td>
</tr>
</tbody>
</table>

3. Select the Protocol. This will specify the format of the headers:
   - Amps (default)
   - Fix
   - NvFix
   - XML
4. Select the **Message Type**. This will specify the format of the data within the message:
   - Fix (default)
   - XML
   - NvFix
   - JSON

5. Select from any of the following **Subscription Modes**:
   - Sow
   - SowAndSubscribe
   - SowAndDeltaSubscribe (default)
   - Subscribe
   - DeltaSubscribe

6. Enter the **Order By Statement** in order to limit the returned data. For example:
   
   /orderDate DESC
   /custumerName ASC

7. Enter any of the following **Option/s** for the selected **Subscription Mode**:
   - cancel
   - live
   - no_empty
   - null
   - no_sowkey
   - oof
   - pause
   - replace
   - resume
   - send_keys
   - timestamp

   **Note:** Leave the **Options** box blank if you selected the **Subscribe** subscription mode.

8. Enter the **Batch Size**. This is the number of messages that will be sent at a time as results are returned. Default is **100**.

9. Enter the **Timeout** for the length of time to wait for the Server response. Default is **5000**.

10. Click . This adds columns to the AMPS connection that will represent sections of the message.

11. Provide the following information:

```plaintext
<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Fix Tag/XPath/Json Path</td>
<td>The Fix Tag/XPath/Json Path of the source schema.</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Type</th>
<th>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Date Format</td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

If **Message Type** is set to **Fix** or **NvFix**, the *Add Column* will display as:

![Add Column Fix/NvFix](image)

If **Message Type** is set to **XML**, the *Add column* will display as:

![Add Column XML](image)

If **Message Type** is set to **JSON**, the *Add Column* will display as:

![Add Column JSON](image)

To delete a column, check its √ or all the column entries, check the topmost √, then click ✗.

12. Instead of manually adding columns, you can opt to click ![Fetch Schema](image) to fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

13. If the **Type** is set to **Text**, it will be listed in the **ID column** drop-down list box and can be used to select a key column to manage data updates and inserts.

   Every message definition needs a Text column to be defined as the **ID column**.

   Furthermore, streaming time series window can be generated by creating a compound key with the ID column, plus a separately specified **Time ID Column**. The **Time ID Column** can be from the source data set, or alternatively automatically generated.

   If the **Time ID Column** is selected, a scrolling time window can be specified.

   ![Time ID Column](image)

   For **Automatic Time Id**, define the **Time Id Column Name**.

   As new data arrives from the subscription, new time slices will automatically be added and the old ones will be deleted.

   If a new ID is received, a new row is added to the in-memory data set representing the AMPS topic subscription. However, if an existing ID is received, an existing row is updated.

14. Click ![Add Source](image). The new data source is added in the **Data Sources** list.
CREATING INFLUXDB INPUT DATA SOURCE

The InfluxDB connector allows for the retrieval of a JSON data set from the InfluxDB. The database communicates over HTTP(S) where you can define a query in the URL to return the desired data.

Steps:

1. In the New Data Source page, select **Input > Influx DB** in the Connector drop-down list.

![InfluxDB Input](image)

2. Enter the following information:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>InfluxDB host address.</td>
</tr>
<tr>
<td>Port</td>
<td>InfluxDB host port. Default is <strong>8086</strong>.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to the InfluxDB service.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to connect to the InfluxDB service.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database that will communicate over the HTTP(S).</td>
</tr>
</tbody>
</table>

3. Enter an SQL-like query language into the Query box.

4. Click **Add**. The new data source is added in the Data Sources list.

CREATING JDBC DATABASE INPUT DATA SOURCE

Steps:

Reference Guide 17.4.0  139
1. In the New Data Source page, select **Input > JDBC** in the Connector drop-down list.

2. You can either select:
   - **JNDI Name**
     Enter the JNDI resource name to be used, then the Username and Password.
   - **URL**
     Enter the URL specific to the database’s JDBC driver, the Driver Class Name specific to the driver, and the Username and Password.

3. Select the appropriate **SQL Dialect** in the drop-down list to be able to generate the correct SQL for the required data repository.
   You can select any of the following SQL dialects: AnsiSQL, Access/Excel, MySQL, Oracle, SQL Server, Sybase IQ/ASA, Sybase ASE, Netezza, Vertica, SQLite, HadoopHive, KxQ, DB2, PostgreSQL, Impala, Redshift, Informix, Teradata, dBase, SparkSQL.
   Default is **AnsiSQL**.

4. Enter the **Timeout**. Default is **60**.

5. Enter an SQL-like query language into the **Query** box.

6. Click **Commit**. The new data source is added in the Data Sources list.

**CREATING A JSON INPUT DATA SOURCE**

The JSON connector allows the retrieval and processing of JSON files, either from a disk, a Text, or from a defined URL.
Steps:

1. In the *New Data Source* page, select **Input > Json** in the *Connector* drop-down list.

2. Select the JSON file source:
   - **File**
     Then enter the JSON File Path.
   - **Text**
     Then enter the text block to be parsed.
   - **Web URL**
     The dialog changes slightly to allow specification of the following:
### PROPERTY | DESCRIPTION
--- | ---
**Path** | The absolute path including the http where the JSON file is located.

**Headers** | - Headers are separated by a comma
- Each Header is entered as Name = Value, where Name and Value can be enclosed in double quotes to allow inclusion of any character except for double quotes
- Name and Value can also be left unquoted, in which case they may not include comma or equals characters

**Content Encoding** | Select the Content Encoding with the HTTP Header: None, GZip, Deflate, or GZip and Deflate

**User Name** | The user name that will be used to connect to the JSON service.

**Password** | The password to connect to the JSON service.

**Http Method** | Select the HTTP Method to map any of the following operations to HTTP requests

<table>
<thead>
<tr>
<th>Http Method</th>
<th>GET, POST, PUT, DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeout</td>
<td>10</td>
</tr>
<tr>
<td>Request Body</td>
<td></td>
</tr>
<tr>
<td>Content Type</td>
<td>application/x-www-form-urlencoded</td>
</tr>
</tbody>
</table>

- GET – retrieve information
- POST – create or update an entity
- PUT – replace an existing entity
- DELETE – remove a request
Reference Guide 17.4.0  143

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Json Path</td>
<td>The Json Path of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is Time.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its ☐ or all the column entries, check the topmost ☐, then click ✖.

5. Instead of manually adding columns, you can opt to click ☀️ to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

6. Click ✗. The new data source is added in the Data Sources list.

**CREATING A KAFKA INPUT DATA SOURCE**

Allows Panopticon Streams to subscribe to Kafka topics on an external cluster.

Steps:

1. In the New Data Source page, select Input > Kafka in the Connector drop-down list.
2. Enter the connection details:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZooKeeper Host</td>
<td>Where the Kafka Server is located.</td>
</tr>
<tr>
<td>ZooKeeper Port</td>
<td>The port number of ZooKeeper being used by the Kafka server. Default is 2181.</td>
</tr>
<tr>
<td>Schema Registry Host</td>
<td>Where the Schema Registry is located. This can be in a different location from the Kafka cluster.</td>
</tr>
<tr>
<td>Schema Registry Port</td>
<td>The port number of the schema registry which provides the serving layer for the metadata. Default is 8081.</td>
</tr>
</tbody>
</table>

3. Click **Fetch Topic** to populate the Topic drop-down list. Initially, the first topic in the list is displayed in the Topic drop-down box.

Click the drop-down list to search and select the desired topic.
4. Check the *From Beginning* box to subscribe from the beginning to the latest messages. If un-checked, you will only be subscribed to the latest messages.

5. Select the Message Type:
   - **Avro**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Enabled</th>
</tr>
</thead>
</table>

   - **Fix**

<table>
<thead>
<tr>
<th>Name</th>
<th>Fix Tag</th>
<th>Type</th>
<th>Date Format</th>
<th>Enabled</th>
</tr>
</thead>
</table>

   - **JSON**

<table>
<thead>
<tr>
<th>Name</th>
<th>JsonPath</th>
<th>Type</th>
<th>Date Format</th>
<th>Enabled</th>
</tr>
</thead>
</table>

   - **Text**

   If **Text** has been selected, confirm the **Decimal Separator**, **Text Qualifier**, **Column Delimiter**, and if the first row of the message includes column headings.

   - **Message Type**: *Text*
   - **Decimal Separator**: .
   - **Text Qualifier**: `<none>`
   - **Column Delimiter**: Comma (`,`)

   - **XML**

<table>
<thead>
<tr>
<th>Name</th>
<th>XPath</th>
<th>Type</th>
<th>Date Format</th>
<th>Enabled</th>
</tr>
</thead>
</table>

6. Select either the dot (.) or comma (,) as the **Decimal Separator**.
NOTES

Prepend 'default:' for the elements falling under default namespace.

7. For non-Avro message types, click + to add columns to the Kafka connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Fix Tag/JsonPath/Text Column Index/XPath</td>
<td>The Fix Tag/JsonPath/Text Column Index/XPath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is Time.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

NOTES

To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them. For example: `yyyy-MM-dd HH:mm:ss.fffffff`

Otherwise, proceed to step 8.

8. Instead of manually adding columns, you can opt to click to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

9. If the Type is selected as Text, it will be listed in the Id Column drop-down list box and can be used to select a key column to manage data updates and inserts.

Note: Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

Furthermore, a streaming time series window can be generated by creating a compound key with the Id Column, plus a separately specified Time ID column. This Time ID column can be from the source dataset, or alternatively automatically generated.

If the Time Id column is selected, then a scrolling time window can be specified.

<table>
<thead>
<tr>
<th>Time Id Column</th>
<th>[Automatic Time Id] ▼</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Id Column Name</td>
<td>Automatic_Timestamp_Column</td>
</tr>
</tbody>
</table>

For Automatic Time Id, define the Time Id Column Name.
As new data arrives from the subscription, new time slices will automatically be added, and old ones will be deleted.

If a new ID is received, a new row is added to the in-memory data set representing the Kafka topic subscription. While if an existing ID is received, an existing row is updated.

10. Click . The new data source is added in the Data Sources list.

CREATING KX KDB+ INPUT DATA SOURCE

The Kx kdb+ input data source allows connection to the Kx kdb+ databases on a polled basis.

Steps:

1. In the New Data Source page, select Input > Kdb+ in the Connector drop-down list.

2. Enter the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Kx kdb+ host address.</td>
</tr>
<tr>
<td>Port</td>
<td>Kx kdb+ host port. Default is 5001.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to Kx kdb+.</td>
</tr>
<tr>
<td>Password</td>
<td>The user name that will be used to connect to Kx kdb+.</td>
</tr>
<tr>
<td>Timeout</td>
<td>30</td>
</tr>
</tbody>
</table>

3. Enter the Timeout. Default is 30.

4. Enter an SQL-like query language into the Query box.

   If a parameter has been defined, the SQL entry can refer to it.
5. Click . The new data source is added in the Data Sources list.

CREATING KX KDB+TICK INPUT DATA SOURCE

The Kx kdb+tick input data source allows connection to a Kx kdb+ ticker plant on a real-time streaming basis.

Specifically, it allows Panopticon Streams to subscribe to Kx kdb+tick through the definition of Service, Table, Symbol, or directly through Functional Subscription.

Steps:
1. In the New Data Source page, select Input > KDB+ Tick in the Connector drop-down list.
2. Enter the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Kx kdb+tick host address.</td>
</tr>
<tr>
<td>Port</td>
<td>Kx kdb+tick host port. Default is 5010.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to Kx kdb+tick.</td>
</tr>
<tr>
<td>Password</td>
<td>The user name that will be used to connect to Kx kdb+tick.</td>
</tr>
</tbody>
</table>

3. Select either Subscription Type:
   - Service
     Enter the following properties:
     - Subscription Name (e.g., sub)
     - Table to subscribe against (e.g., trade)
     - Symbol to subscribe against (e.g., AAPL)

   **NOTES**
   Multiple symbols should be separated by a comma.

   - Functional Subscription
     Enter the functional subscription that needs to be issued (e.g., .u.sub['trade;'])

4. Click Fetch Schema to retrieve the schema of the configured subscription. This populates the Id Column with the set of columns from the schema of type sym and the text array such as Character/Boolean/GUID, etc.

5. Check Constrain subscription to matching symbols to select the column which contains specific symbols. Otherwise, the filtering against these symbols will not take place.

   **NOTES**
   The Constrain subscription to matching symbols only lists sym fields. Therefore, if you select a non sym type in the Id Column, it is not recommended to select the default value [Id Column] in the Constrain subscription to matching symbols drop-down list.

6. Activate or deactivate Initialize with historic data. If unchecked, the data source will only be populated with streaming updates that are subscribed against. If checked, the data source is first initialized against a store of data, after which subscribed streaming updates are then applied.
7. Enter the following information:
   • Host
   • Port
   • User Name
   • Password
   • Query

NOTES

These entries can be parameterized.

8. Check Deferred Sync Query box to allow the Kxkdb+tick data source to support synchronous and asynchronous reads. The advantage of using this option is that there is no queue on the Kx kdb+tick server side, queries are farmed out to slaves and returned to asynchronous instead.

The {Query} parameter is used as a place holder for the target query that is defined in the Query builder.

9. Select the Flatten List Limit.
   This allows retrieval of the first ‘n’ items in the list and produce new columns in the output schema with a dot notation.

   For example, if there are two nested fields (BidPrices and OfferPrices) and the flatten list limit selected is five, then the output schema will be:

   BidPrices.1, BidPrices.2, BidPrices.3, BidPrices.4, BidPrices.5, OfferPrices.1, OfferPrices.2, OfferPrices.3, OfferPrices.4, OfferPrices.5

   If there are less than five items in the list, then the values will be null.

NOTES

Currently, this feature works for the Service subscription type. Also, it only flattens numeric columns.

10. Select the ID Column.
    This defines the column that uniquely identifies a row in the returned stream, and is used for processing inserts, updates and deletes.

    This allows the latest update time and its age to be highlighted by the defined color range in the output dashboard.

    Furthermore, a streaming time series window can be generated by creating a compound key with the Id Column, plus a separately specified Time ID column. This Time ID column can be from the source dataset, or alternatively automatically generated.

    If the Time Id column is selected, then a scrolling time window can be specified.
For **Automatic Time Id**, define the **Time Id Column Name**.

As new data arrives from the subscription new time slices will automatically be added, and old ones will be deleted.

11. Click . The new data source is added in the *Data Sources* list.

**CREATING MQTT INPUT DATA SOURCE**

The MQTT connector allows connection to MQTT’s message bus on a real-time streaming basis. Specifically, the connector allows Panopticon Streams to subscribe to FIX, JSON, Text or XML based messages that are published on particular topics. The data format itself is arbitrary, and as a consequence, the connection includes the message definition.

**Steps:**

1. In the *New Data Source* page, select **Input > MQTT** in the *Connector* drop-down list.

2. Enter the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broker</td>
<td>The location of the message broker.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to MQTT.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that will be used to connect to MQTT.</td>
</tr>
</tbody>
</table>
3. Select the **Message Type**. This will specify the format of the data within the message:
   - **Fix**
   
   ![Fix Table]

   - **JSON**
   
   ![JSON Table]

   - **Text**

   If **Text** has been selected, confirm the **Decimal Separator**, **Text Qualifier**, **Column Delimiter**, and if the first row of the message includes column headings.

   ![Text Qualifier]

   ![Column Delimiter]

   ![First Row Headings]

   ![Column Index Table]

4. Select either the dot (.) or comma (,) as the **Decimal Separator**.

   **NOTES**

   Prepend 'default:' for the elements falling under default namespace.

5. Click **+** to add columns to the Solace connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>XPath/JsonPath/Fix Tag/Column Index</td>
<td>The XPath/JsonPath/Fix Tag/Column Index of the source schema.</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Date Format</strong></td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td><strong>NOTE:</strong></td>
<td>To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them.</td>
</tr>
<tr>
<td></td>
<td>For example: <code>yyyy-MM-dd HH:mm:ss.ffffff</code></td>
</tr>
<tr>
<td><strong>Enabled</strong></td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its [x] or all the column entries, check the topmost [x], then click ✖.

6. Instead of manually adding columns, you can opt to click [Fetch Schema] to fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

7. If the **Type** is selected as **Text**, it will be listed in the **Id Column** drop-down list box and can be used to select a key column to manage data updates and inserts.

   **Note:** Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

Furthermore, a streaming time series window can be generated by creating a compound key with the **Id Column**, plus a separately specified **Time ID** column. This **Time ID** column can be from the source dataset, or alternatively automatically generated.

If the **Time Id column** is selected, then a scrolling time window can be specified.

For **Automatic Time Id**, define the **Time Id Column Name**.

As new data arrives from the subscription new time slices will automatically be added, and old ones will be deleted.

If a new ID is received, a new row is added to the in-memory data set representing the ActiveMQ topic subscription. While if an existing ID is received, an existing row is updated.

8. Click [Add]. The new data source is added in the **Data Sources** list.

### CREATING MS EXCEL INPUT DATA SOURCE

This is the most commonly used data connector when prototyping and is used for retrieving data from MS Excel workbooks or spreadsheets, where for each selected sheet, the first row contains the field/column names, and subsequent rows contain the data.

**Steps:**

1. In the **New Data Source** page, select **Input > MS Excel** in the **Connector** drop-down list.
2. Enter the *Excel File Path* of the source file.
   
   **Example:** E:\ bidoffertrade.xls

3. Click **Fetch Sheets**. This will populate the Select Sheet drop-down list box.

4. Select the required sheet.

5. Click ![icon]. The new data source is added in the *Data Sources* list.

**CREATING ONETICK CLOUD INPUT DATA SOURCE**

The OneTick Cloud connector allows access to historic market data with no software dependencies by using the OneTick Cloud and their web API.

**Steps:**

1. In the *New Data Source* page, select **Input > OneTick Cloud** in the *Connector* drop-down list.

2. Enter the OneTick Cloud WebAPI URL into the *Query URL* box with the following form:
Where:

- \( s, e, \) timezone – the start and end time of the query \( \text{YYYYMMDDhhmmss} \) form. The timezone used to interpret this value is taken from the timezone parameter.
- response – the supported response format is csv.
- compression – if available, this option enables gzip compression of the results stream. Large data should always be pulled with compression on.

3. Enter the Username (email) and Password to execute the query and retrieve the data. Note that the Username is case sensitive.

4. Enter the time window Start Date and End Date.

5. Enter the Symbol List. This value filters the query output with matching symbols.

   To make it work, ensure to include Symbol in the Query URL. Consequently, the data will be filtered out for the input (Symbols) provided in the Symbol List field.

6. Enter the Symbol Pattern. This value filters the query output with the data for all the symbols with matching pattern.

   To make it work, ensure to include Symbol_Pattern in the Query URL. Consequently, the data will be filtered (for all the Symbols) with matching pattern provided in the Symbol Pattern field.

7. Click \( + \). A new column entry displays. Enter or select the following properties:

    | PROPERTY      | DESCRIPTION                                      |
    |---------------|--------------------------------------------------|
    | Name          | The column name of the source schema.            |
    | Column Index  | The column index controls the position of a column. Must be \( \geq 0 \). |
    | Type          | The data type of the column. Can be a Text, Numeric, or Time |
    | Date Format   | The format when the data type is Time.           |
    | Enabled       | Determines whether the message should be processed. |

   To delete a column, check its \( \square \) or all the column entries, check the topmost \( \square \), then click \( - \).

8. Instead of manually adding columns, you can opt to click \( \square \) to fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first \( n \) rows of the input data source.

9. The time zone of input parameters and output data is by default unchanged. Changing the time zone is supported through the Timezone list box, based on the assumption that data is stored in UTC time and outputs are presented in the selected time zone.
10. Click . The new data source is added in the Data Sources list.

**CREATING RABBITMQ INPUT DATA SOURCE**

The RabbitMQ connector allows connection to RabbitMQ’s message bus on a real-time streaming basis. Specifically, the connector allows Panopticon Streams to subscribe to XML, JSON, Text or FIX based messages that are published on particular topics.

**Steps:**

1. In the New Data Source page, select **Input > RabbitMQ** in the Connector drop-down list.

   ![RabbitMQ Input](image)

   **PROPERTY** | **DESCRIPTION**
   --- | ---
   Broker | The location of the message broker.
   User Name | The user name that will be used to connect to RabbitMQ.

   2. Enter the connection details including:
3. Select any of the following Exchange Types:

<table>
<thead>
<tr>
<th>EXCHANGE TYPE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>A direct exchange with no name that is pre-declared by the broker. Selecting this exchange type disables the Exchange section (Exchange and Routing Key properties).</td>
</tr>
<tr>
<td>Fanout</td>
<td>Broadcasts all of the messages it receives to all of the queues it knows and the routing key is ignored (the Routing Key field is disabled).</td>
</tr>
<tr>
<td>Direct</td>
<td>Delivers messages to queues based on a message routing key. It is ideal for the unicast routing of messages, although it can be used for multicast routing as well.</td>
</tr>
<tr>
<td>Topic</td>
<td>A message sent with a particular routing key will be delivered to all of the queues that are bound with a matching binding key.</td>
</tr>
<tr>
<td>Headers</td>
<td>Exchanges routed based on arguments containing headers and optional values.</td>
</tr>
</tbody>
</table>

4. Depending on the selected Exchange Type, select or define the following:

<table>
<thead>
<tr>
<th>EXCHANGE TYPE PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchange</td>
<td>Name of the exchange.</td>
</tr>
<tr>
<td>Durable</td>
<td>Enable so the exchange can survive a broker restart.</td>
</tr>
<tr>
<td>Auto Delete</td>
<td>Enable so the exchange is deleted when the last queue is unbound from it.</td>
</tr>
<tr>
<td>Routing Key</td>
<td>The routing key used to deliver messages to queues.</td>
</tr>
<tr>
<td>Headers</td>
<td>This field is only available when the message type is <strong>Header</strong>. Binding a queue to a Headers exchange is possible using more than one header for matching. Setting <code>x-match</code> to <strong>any</strong>, means just one matching value is sufficient. Setting it to <strong>all</strong> means that all values must match. Default is <strong>x-match=all</strong>.</td>
</tr>
</tbody>
</table>

5. Check the *Explicit Queue* box and enter the custom queue name. Then enter or enable the following properties:
## QUEUE PROPERTY | DESCRIPTION

<table>
<thead>
<tr>
<th>Properties</th>
<th>The custom queue property.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Durable</td>
<td>Enable so the queue can survive a broker restart.</td>
</tr>
<tr>
<td>Auto Delete</td>
<td>Enable so the queue that had the least consumer will be deleted when that connection closes.</td>
</tr>
</tbody>
</table>

6. Select the **Message Type**:
   - **FIX**
     - Name
     - Fix Tag
     - Type
     - Date Format
     - Enabled

   - **JSON**
     - Name
     - JsonPath
     - Type
     - Date Format
     - Enabled

   - **Text**
     - Message Type
     - Decimal Separator
     - Text Qualifier
     - Column Delimiter
     - First Row Headings
     - Column Index controls the position of a column, Must be >= 0.

   - **XML**
     - Name
     - XPath
     - Type
     - Date Format
     - Enabled

7. Select either the dot (.) or comma (,) as the **Decimal Separator**.

### NOTES

Prepend 'default:' for the elements falling under default namespace.

8. Click + to add columns to the RabbitMQ connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
</table>

---

158  Reference Guide 17.4.0
<table>
<thead>
<tr>
<th>Name</th>
<th>The column name of the source schema.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fix Tag/Json Path/Text Column Index/Xpath</td>
<td>The Fix Tag/Json Path/Text Column Index/Xpath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td><strong>NOTE:</strong></td>
<td>To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them.</td>
</tr>
<tr>
<td></td>
<td>For example: <strong>yyyy-MM-dd HH:mm:ss.fffffff</strong></td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its or all the column entries, check the topmost , then click .

9. Instead of manually adding columns, you can opt to click to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

10. If the **Type** is selected as **Text**, it will be listed in the **Id Column** drop-down list box and can be used to select a key column to manage data updates and inserts.

    **Note:** Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

    Furthermore, a streaming time series window can be generated by creating a compound key with the **Id Column**, plus a separately specified **Time ID column**. This **Time ID column** can be from the source dataset, or alternatively automatically generated.

    If the **Time Id column** is selected, then a scrolling time window can be specified.

    For **Automatic Time Id**, define the **Time Id Column Name**.

    As new data arrives from the subscription new time slices will automatically be added, and old ones will be deleted.

    If a new ID is received, a new row is added to the in-memory data set representing the ActiveMQ topic subscription. While if an existing ID is received, an existing row is updated.

11. Click . The new data source is added in the **Data Sources** list.

**CREATING SOLACE INPUT DATA SOURCE**

The Solace connector allows connection to Solace’s message bus on a real time streaming basis. Specifically, the connector allows Panopticon Streams to subscribe to messages that are published in particular topics in Solace and consequently, perform operational analytics.
Steps:

1. In the *New Data Source* page, select **Input > Solace** in the *Connector* drop-down list.

2. Enter the connection details including:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Solace host address.</td>
</tr>
<tr>
<td>VPN Name</td>
<td>Message VPN name. Default is <strong>default</strong>.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to Solace.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that will be used to connect to Solace.</td>
</tr>
</tbody>
</table>

3. Enter the *Topic* or the queue physical name.

4. Select the *Message Type*. This will specify the format of the data within the message:
   - **SDTMap**
   - **JSON**
   - **Text**
If **Text** has been selected, confirm the **Decimal Separator, Text Qualifier, Column Delimiter**, and if the first row of the message includes column headings.

- **XML**

5. Select either the dot (.) or comma (,) as the **Decimal Separator**.

**NOTES**

Prepend 'default:' for the elements falling under default namespace.

6. Click + to add columns to the Solace connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Type/JsonPath/Column Index/XPath</td>
<td>The SDTMap Type/JsonPath/Text Column Index/XPath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong></td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td><strong>NOTE:</strong></td>
<td>To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them. For example: <code>yyyy-MM-dd HH:mm:ss.ffffff</code></td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its ☐ or all the column entries, check the topmost ☐, then click −.
7. Instead of manually adding columns, you can opt to click ![image] to fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first 'n' rows of the input data source.

8. If the Type is selected as **Text**, it will be listed in the **Id Column** drop-down list box and can be used to select a key column to manage data updates and inserts.

   **Note:** Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

   Furthermore, a streaming time series window can be generated by creating a compound key with the **Id Column**, plus a separately specified **Time ID** column. This **Time ID** column can be from the source dataset, or alternatively automatically generated.

   If the **Time Id column** is selected, then a scrolling time window can be specified.

   For **Automatic Time Id**, define the **Time Id Column Name**.

   As new data arrives from the subscription, new time slices will automatically be added, and old ones will be deleted.

   If a new ID is received, a new row is added to the in-memory data set representing the ActiveMQ topic subscription. While if an existing ID is received, an existing row is updated.

9. Click ![image]. The new data source is added in the **Data Sources** list.

**CREATING STREAM SIMULATOR INPUT DATA SOURCE**

The Stream Simulator connector is very similar to the Text connector with the addition of the time windowing of message queue connectors.

Creating the Stream Simulator input data source includes setting for how fast and how many messages are pushed through in each batch.

**Steps:**

1. In the **New Data Source** page, select **Input > Stream Simulator** in the **Connector** drop-down list.
2. Select the text file source:
   - **Text**
     
     Enter the text block to be parsed.
   
   - **File**
     
     Ensure to enter the *Text File Path* of the source file.
The dialog changes slightly to allow specification of the following:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>The location of the message broker.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to Stream Simulator.</td>
</tr>
<tr>
<td>Password</td>
<td>The user name that will be used to connect to Stream Simulator.</td>
</tr>
<tr>
<td>Request Body</td>
<td>The Request Body for HTTP POST.</td>
</tr>
<tr>
<td>Timeout</td>
<td>The length of time to wait for the server response (10 to 300). Default is 10.</td>
</tr>
</tbody>
</table>

The standard settings controlling how the text file is parsed, is listed. These include:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skip First N Rows</td>
<td>Specifies the number of rows that will be skipped.</td>
</tr>
<tr>
<td>Data Type Discovery</td>
<td>Specifies how many rows from the text file should be used when automatically determining the data types of the resulting columns.</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>Specifies if fields are enclosed by text qualifiers, and if present to ignore any column delimiters within these text qualifiers.</td>
</tr>
<tr>
<td>Column Delimiter</td>
<td>Specifies the column delimiter to be used when parsing the text file.</td>
</tr>
<tr>
<td>First Row Headings</td>
<td>Determines if the first row should specify the retrieved column headings, and not be used in data discovery.</td>
</tr>
</tbody>
</table>

3. Click +. A new column entry displays. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Column Index</td>
<td>The column index controls the position of a column. Must be &gt;= 0.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
</tbody>
</table>
Date Format | The format when the data type is Time.
---|---
Enabled | Determines whether the message should be processed.

To delete a column, check its ☑ or all the column entries, check the topmost ☑, then click -.

4. Instead of manually adding columns, you can opt to click - to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

6. Enter the **Update Set Size** and **Start Up Set Size**.

7. Select the **ID Column**.

   This defines the column that uniquely identifies a row in the returned stream, and is used for processing inserts, updates and deletes.

   By default, only the latest data will be loaded into memory. A streaming time series window can be generated by creating a compound key with the **Id Column**, plus a separately specified **Time ID column**. This Time ID column can be from the source dataset, or alternatively automatically generated.

   If the Time Id column is selected, then a scrolling time window can be specified. As new data arrives from the subscription new time slices will automatically be added, and old ones will be deleted.

   ![Time Id Column](image)

   For **Automatic Time Id**, define the **Time Id Column Name**.

8. Check the **Loop** box to enable looping through the file.

9. Click - . The new data source is added in the **Data Sources** list.

### CREATING TEXT INPUT DATA SOURCE

The Text connector allows the retrieval and processing of delimited Text files (such as CSV, TSV, and so on), either from a disk or from a defined URL.

**Steps:**

1. In the **New Data Source** page, select **Input > Text** in the **Connector** drop-down list.
2. Select the text file source:
   - **Text**
     Enter the text block to be parsed.
     ```plaintext
     Text File Source: Text
     Text
     ```
   - **File**
     Ensure to enter the *Text File Path* of the source file.
     ```plaintext
     Text File Source: File
     Text File Path
     ```
   - **WebURL**
The dialog changes slightly to allow specification of the following:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>The absolute path including the http where the Text file is located.</td>
</tr>
<tr>
<td>Headers</td>
<td>• Headers are separated by a comma</td>
</tr>
<tr>
<td></td>
<td>• Each Header is entered as <strong>Name = Value</strong>, where Name and Value can be</td>
</tr>
<tr>
<td></td>
<td>enclosed in double quotes to allow inclusion of any character except for</td>
</tr>
<tr>
<td></td>
<td>double quotes</td>
</tr>
<tr>
<td></td>
<td>• Name and Value can also be left unquoted, in which case they may not</td>
</tr>
<tr>
<td></td>
<td>include comma or equals characters</td>
</tr>
<tr>
<td>Content Encoding</td>
<td>Select the <strong>Content Encoding</strong> with the HTTP Header: <strong>None</strong>, <strong>GZip</strong>,</td>
</tr>
<tr>
<td></td>
<td><strong>Deflate</strong>, or <strong>GZip and Deflate</strong></td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to the Text service.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to connect to the Text service.</td>
</tr>
<tr>
<td>Http Method</td>
<td>Select the HTTP Method to map any of the following operations to HTTP</td>
</tr>
<tr>
<td></td>
<td>requests</td>
</tr>
<tr>
<td></td>
<td>• <strong>GET</strong> – retrieve information</td>
</tr>
<tr>
<td></td>
<td>• <strong>POST</strong> – create or update an entity</td>
</tr>
<tr>
<td></td>
<td>• <strong>PUT</strong> – replace an existing entity</td>
</tr>
<tr>
<td></td>
<td>• <strong>DELETE</strong> – remove a request</td>
</tr>
</tbody>
</table>
Timeout | The length of time to wait for the server response (10 to 300). Default is 10.
--- | ---
Request Body | The Request Body for HTTP POST.
Content Type | The required Content Type. Default is application/x-www-form-urlencoded

The standard settings controlling how the text file is parsed, is listed. These include:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skip First N Rows</td>
<td>Specifies the number of rows that will be skipped.</td>
</tr>
<tr>
<td>Data Type Discovery</td>
<td>Specifies how many rows from the text file should be used when automatically determining the data types of the resulting columns.</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>Specifies if fields are enclosed by text qualifiers, and if present to ignore any column delimiters within these text qualifiers.</td>
</tr>
<tr>
<td>Column Delimiter</td>
<td>Specifies the column delimiter to be used when parsing the text file.</td>
</tr>
<tr>
<td>First Row Headings</td>
<td>Determines if the first row should specify the retrieved column headings, and not be used in data discovery.</td>
</tr>
</tbody>
</table>

3. Click +. A new column entry displays. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>Column Index</td>
<td>The column index controls the position of a column. Must be ( \geq 0 ).</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time.</td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is Time.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its \( \) or all the column entries, check the topmost \( \), then click -.

4. Instead of manually adding columns, you can opt to click \( \) to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first 'n' rows of the input data source.

5. Click . The new data source is added in the Data Sources list.

**CREATING WEBSOCKET INPUT DATA SOURCE**

The WebSocket connector is very similar to the Stream Simulator connector, except that rather than looping through a file, it would either connect through web sockets, long polling, or repeatedly poll an external URL for new records to process.

Steps:

---

168 Reference Guide 17.4.0
1. In the New Data Source page, select **Input > WebSocket** in the **Connector** drop-down list.

![WebSocketInput](image)

2. Enter the connection details:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>The URL to which the WebSocket server will respond to.</td>
</tr>
<tr>
<td>User ID</td>
<td>The User ID that will be used to connect to the WebSocket server.</td>
</tr>
<tr>
<td>Password</td>
<td>The password that will be used to connect to the WebSocket server.</td>
</tr>
<tr>
<td>Request Body</td>
<td>For both the HTTP and ws:// POST requests sent to the WebSocket server.</td>
</tr>
<tr>
<td>Timeout</td>
<td>The length of time to wait for the server response (10 to 300). Default is 10.</td>
</tr>
</tbody>
</table>

3. Enter the **Record Path**.

4. Select the Message Type:
   - **JSON**
   - **Text**

   If **Text** has been selected, confirm the **Decimal Separator, Text Qualifier, Column Delimiter**, and if the first row of the message includes column headings.
5. Select either the dot (.) or comma (,) as the **Decimal Separator**.

**NOTES**

Prepend `default:` for the elements falling under default namespace.

6. Click + to add columns to the WebSocket connection that represent sections of the message. Then enter or select:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>JsonPath/Text Column Index/XPath</td>
<td>The JsonPath/Text Column Index/XPath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a <strong>Text</strong>, <strong>Numeric</strong>, or <strong>Time</strong>.</td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is <strong>Time</strong>.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message field should be processed.</td>
</tr>
</tbody>
</table>

**NOTES**

To parse and format times with higher than millisecond precision, the format string needs to end with a period followed by sequence of lower case Fs. There can be no additional characters following them.

For example: **yyyy-MM-dd HH:mm:ss.fffffff**
7. Instead of manually adding columns, you can opt to click to the fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

8. If the **Type** is selected as **Text**, it will be listed in the **Id Column** drop-down list box and can be used to select a key column to manage data updates and inserts.

   **Note:** Every message definition needs a text column to be defined as the ID column. By default, only the latest data will be loaded into memory.

   Furthermore, a streaming time series window can be generated by creating a compound key with the **Id Column**, plus a separately specified **Time ID** column. This **Time ID** column can be from the source dataset, or alternatively automatically generated.

   If the **Time Id column** is selected, then a scrolling time window can be specified.

   - **Time Id Column**: [Automatic Time Id]
   - **Time Id Column Name**: Automatic_Timestamp_Column

   For **Automatic Time Id**, define the **Time Id Column Name**.

   As new data arrives from the subscription, new time slices will automatically be added, and old ones will be deleted.

   If a new ID is received, a new row is added to the in-memory data set representing the WebSocket topic subscription. While if an existing ID is received, an existing row is updated.

9. Click . The new data source is added in the **Data Sources** list.

**CREATING XML INPUT DATA SOURCE**

The XML connector allows the retrieval and processing of XML files, either from a disk, a Text, or from a defined URL.
Steps:

1. In the New Data Source page, select Input > Xml in the Connector drop-down list.

![XML Input Diagram]

2. Select the XML file source:
   - **File**
     Ensure to enter the XML File Path of the source file.

   ![XML File Source Diagram]

   - **Text**
     Then enter the text block to be parsed.

![XML Text Source Diagram]
• **Web URL**

The dialog changes to allow specification of the following:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>The absolute path including the http where theXml file is located.</td>
</tr>
<tr>
<td>Headers</td>
<td>• Headers are separated by a comma</td>
</tr>
<tr>
<td></td>
<td>• Each Header is entered as <strong>Name = Value</strong>, where <strong>Name</strong> and <strong>Value</strong></td>
</tr>
<tr>
<td></td>
<td>can be enclosed in double quotes to allow inclusion of any character except</td>
</tr>
<tr>
<td></td>
<td>for double quotes</td>
</tr>
<tr>
<td></td>
<td>• <strong>Name</strong> and <strong>Value</strong> can also be left unquoted, in which case they</td>
</tr>
<tr>
<td></td>
<td>may not include comma or equals characters</td>
</tr>
<tr>
<td>Content Encoding</td>
<td>Select the <strong>Content Encoding</strong> with the HTTP Header: <strong>None, GZip, Deflate,</strong></td>
</tr>
<tr>
<td></td>
<td>or <strong>GZip and Deflate</strong></td>
</tr>
<tr>
<td>User Name</td>
<td>The user name that will be used to connect to the Xml service.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to connect to the Xml service.</td>
</tr>
<tr>
<td>Http Method</td>
<td>Select the <strong>HTTP Method</strong> to map any of the following operations to HTTP</td>
</tr>
<tr>
<td></td>
<td>requests</td>
</tr>
<tr>
<td></td>
<td>• <strong>GET</strong> – retrieve information</td>
</tr>
<tr>
<td></td>
<td>• <strong>POST</strong> – create or update an entity</td>
</tr>
<tr>
<td></td>
<td>• <strong>PUT</strong> – replace an existing entity</td>
</tr>
<tr>
<td></td>
<td>• <strong>DELETE</strong> – remove a request</td>
</tr>
</tbody>
</table>
Timeout | The length of time to wait for the server response (10 to 300). Default is 10.
--- | ---
Request Body | The Request Body for HTTP POST.
Content Type | The required Content Type. Default is application/x-www-form-urlencoded

3. Enter the Record XPath (e.g., //myroot/items/item).

4. Click +. A new column entry displays. Enter or select the following properties:

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The column name of the source schema.</td>
</tr>
<tr>
<td>XPath</td>
<td>The XPath of the source schema.</td>
</tr>
<tr>
<td>Type</td>
<td>The data type of the column. Can be a Text, Numeric, or Time</td>
</tr>
<tr>
<td>Date Format</td>
<td>The format when the data type is Time.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Determines whether the message should be processed.</td>
</tr>
</tbody>
</table>

To delete a column, check its \( \square \) or all the column entries, check the topmost \( \square \), then click \( \text{-} \).

5. Instead of manually adding columns, you can opt to click \( \text{\linebreak} \) to fetch the schema based on the connection details. This populates the list of columns with the data type found from inspecting the first ‘n’ rows of the input data source.

6. Click \( \text{\linebreak} \). The new data source is added in the Data Sources list.
Modifying Data Sources

Steps:

1. On the **Data Sources** tab, click the link of a data source you want to modify. The corresponding data source page is displayed.

   ![Data Source Page](image)

   All of the controls that are editable can be modified.

2. Make the necessary changes then click the icon. The context menu displays with two saving options:

   - ✷ Save
     
     Click to save the changes made in the data source.

   - ✷ Save as Copy
     
     Click to make a duplicate of the data source. The original name is appended with _Copy_.

     To change the **Data Source Name**, click on it to make it editable, then enter a new one and click ✅.

Other Data Sources Operations

On the **Data Sources** tab, you can also perform the following:

- Sort the list
SORTING THE LIST OF DATA SOURCES

By default, the list of data sources is sorted by Name in an ascending order. You can modify the sorting of the list by clicking the ▼ or ▲ button of the Name, Last Modified, or Plugin columns. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.

VIEWING THE APPLICATION USAGES

On the Data Sources tab, you can view the applications that currently use a data source.

Steps:

1. Click the □ icon of an application data source.
   The list of applications that currently use the data source displays.
2. Click Close.

DOWNLOADING AN APPLICATION DATA SOURCE

Click the ‹ of an application data source to download and save a copy.

DELETING A DATA SOURCE

Steps:

1. Click the ✗ icon.
   A confirmation message displays with the list of applications that will be impacted upon deletion.
2. Click Yes to delete.
SEARCHING FOR DATA SOURCES

To search for a particular data source, enter it in the Filter Sources box.

You can also enter one of more characters into the Filter Data Sources box and the suggested list of data sources that matched the entries will be displayed.

When an application has been started, the data producers used to generate data from the data sources will be displayed on the **Data Producer** tab where you can:

- Refresh data producers
- Start or Stop data producers

![Data Producer tab](image)

Figure 9-1. Data Producer tab. Initially, no records are displayed when there are no running applications or the applications that are running have no data producers

![Data Producer tab with data producers](image)

Figure 9-2. Data Producer tab with data producers currently started
REFRESH DATA PRODUCERS

Steps:

1. On the Data Producers tab, click the Refresh icon of a data producer. A confirmation message displays.
2. Click Yes.

STARTING OR STOPPING DATA PRODUCERS

To start a Data Producer:

1. Click . A confirmation message displays.
2. Click Yes. The icon changes to .

To stop the Data Producer:

1. Click . A confirmation message displays.
2. Click Yes. The icon changes to .
[10] Monitoring Engine Metrics and Application Topics

The Monitoring tab provides the ability to monitor the engine metrics that can help determine which part of the application is causing data bottlenecks, among others.

![Monitoring and Application Topics tab](image)

### ENGINE METRIC | DESCRIPTION
--- | ---
Free Physical Memory | The amount of free physical memory available to the Panopticon Streams server.
Total Physical Memory | The total amount of physical memory.
Total JVM CPU Usage | The recent CPU usage for the Java Virtual Machine process.
Outgoing Kafka Network bytes/sec | The number of bytes per second that is produced by Kafka.
Incoming Kafka Network bytes/sec | The number of bytes per second that is consumed by Kafka.
<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purgatory Queue Size Producer</td>
<td>The number of requests waiting to send the topic data.</td>
</tr>
<tr>
<td>Purgatory Queue Size Fetch</td>
<td>The number of requests waiting to fetch the topic data.</td>
</tr>
</tbody>
</table>

It also displays the list of input and output topics currently running.

![Image](image-url)

**Figure 10-2.** Monitoring tab when disconnected to the engine
Managing Topics

While running or executing an application, input and output topics are retrieved and displayed on the **Monitoring** tab.

You can perform the following:

- View and monitor the number of retrieved messages and the number of retrieved messages per second
- Define a filter among the topics
Sort the list of topics

FILTER TOPICS

The topics can be filtered by entering letters, numbers, or underscores in the Topic or Application text box.
For the Type of application, enter a text (either Output or Input) into the text box above the listing.

SORTING THE LIST OF TOPICS

You can modify the sorting of the list by clicking the or button of the Topic, Type, Application, #Messages, or #Messages/sec column. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.

MOVING TO OTHER TOPICS LIST PAGES

Go to the other topics pages by clicking:
- any link of a page number
- . This displays the previous page
- . This displays the next page
Scheduling Tasks

Panopticon Streams supports scheduling of tasks such as daily deletion of application topics.

Figure 11-1. Scheduler tab

Create Task to Clear Topic Data

Steps:
1. Select the Scheduler tab.
2. Enter the Name of the task. Ensure the name is unique.
3. Tap the Activated slider to turn it on.
4. Select the Trigger. You can either select:
5. Period then enter the Interval (in seconds), or
7. **CRON** then enter the **CRON Expression**

5. Select the task **Type**: **Clear Topic Data**.
6. Enter the **Description** of the task.

   Upon selecting **Clear Topic Data**, the **Scheduler** tab changes to allow selection of the application.
7. Select the *Application* in the drop-down list. These are the applications available on the *Applications* tab.

8. Click **Create**. The new task is added in the list.

![Create Task Interface](image)

A task displays the following columns: *Name, Status, Type, Trigger, Created By, Created, and Last Run*

You can modify the sorting of the list by clicking the ▼ or ▲ button of any of these columns. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.

Tasks can also be:

- manually started
  
  Instead of waiting for the set Period interval or CRON Expression, you can manually execute the task by clicking **. A confirmation message displays. Click **YES**.

- **modified** or **duplicated**

- **deleted**

  Click **X** of a task. A confirmation message displays. Click **YES**.
Modify a Scheduled Task

Steps:
1. On the **Scheduler** tab, click the link of a task to modify. The properties of the task are displayed.

![Scheduler screenshot]

2. Apply the desired changes.
3. Click **Update**.
Create a Duplicate of a Scheduled Task

Steps:

1. On the **Scheduler** tab, click the link of a task to make a duplicate copy. The properties of the task are displayed.
2. Apply the desired changes, making sure to enter a new name.
3. Click **Create as New**. A duplicate of the task is created and added in the **Scheduled Tasks** list.
[12] Logging/Monitoring

View Logs

View the latest 300 rows of a *Logging Level* on the **Logs** tab:

- **FINEST** (lowest level)
- **FINER**
- **FINE**
- **CONFIG**
- **INFO** (default level)
- **WARNING**
- **SEVERE** (highest level)

![Altair Panopticon: Streams](image)

*Figure 12-1. Logs tab*
ZooKeeper and Kafka produce data and log directories in the `<Streams-properties#cep.kafka.path>/Data`
- /logs
- /tmp/kafka-logs
- /Zookeeper/* (the ZooKeeper output files)

Steps:

1. Click the **Logs** tab. Initially, the default level (INFO) logs are displayed.

   ![Logs tab in Panopticon Streams](image)

   - Logs level: INFO
   - Log entries showing various levels of output

2. Select another **Logging Level** in the drop-down.
   
   For example: FINEST
The latest 300 rows of the selected log level or higher are fetched.

3. You can also click any of the following buttons:

- to pause the logging, it changes to
• to resume the logging
• to copy log to clipboard
• Clear all to clear the logs

SET FILE LOGGING LEVEL

On the System tab, the level that is logged to file can be set.
Steps:

1. The current set level (e.g., **WARNING**) is displayed. To change, click the drop-down list and select another log level.
The new log level is written in the `Streams.properties` file:

```ini
logger.level.file=FINEST
```

The **Parameters** tab supports adding, modifying, and deleting global parameters that will pull and enter specific data into the different components of an application model.

![Parameters tab](image)

**Adding Parameters**

Steps:

1. On the **Parameters** tab, click **Add Parameter**.
   
   A new parameter entry displays.

2. Enter a **Name** for the new parameter and the **Value**.

3. Check the **Encrypted** box to encrypt the value.
4. Click **OK**. The new parameter is added in the list.

New parameters are added in the `DefaultParameters.xml` file located in the `App_data` folder (i.e., `c:\streamsseverdata`).

Modifying Parameters

Steps:

1. On the **Parameters** tab, click the **Edit** link of a parameter you want to modify.
   The **Name**, **Value**, and **Encrypted** controls are enabled.

2. Make the necessary changes then click **OK**.
Deleting Parameters

Steps:

1. On the Parameters tab, click the Remove link of a parameter you want to delete. A confirmation message displays.
2. Click YES to delete.

REFRESH PARAMETERS

Click Refresh to refresh the values that are being pulled by the application models.

SORTING THE LIST OF PARAMETERS

By default, the parameters are listed based on the sequence that they were added. You can modify the sorting of the list by clicking the ▼ or ▲ button of the Name, Value, or Encrypted columns. The icon beside the column that was used for the sorting will indicate if it was in an ascending or descending order.
[14] Example Applications

The Panopticon Streams is installed with a series of example applications:

- **AggregationExample** – Demonstrates how to aggregate data based on a grouping key and a set of aggregated fields.
  
  Includes simple aggregations such as avg, count, first, last, max, min, samples, sum, sdevp, sdevs, Sum, varp, and vars.

- **BranchExample** – Demonstrates how to split a stream into one or more branches.

- **CalculateRemoveReplaceNull** – Demonstrates how to:
  - remove and replace fields from output schemas
  - set a field value to null
  - set a field value to the current timestamp

- **CalculationExample** – Includes the SquareRoot calculation.

- **CalculationsExample** – Includes the following calculations:
  - Numeric calculations such as Abs, SquareRoot, Subtract, Multiply, Divide, Truncate, IF
  - Text calculations such as Upper, Lower, Proper, Left, Right, Mid, Concat, Find
  - Time Period calculations such as DateDiff

  In addition, data type casting between Text, Number, and Date/Time

- **ConflateExample** – Demonstrates how to lower the frequency of updates by setting a fixed interval.

- **EmailExample** – Shows how to send an email via SMTP where the SMPT and email settings can be parameterized. Each record passed to the connector results in an email which can be primarily used as an output for alerting, having a conditional expression that would need to be fulfilled for a record to be forwarded to the output.

  Requires the EmailWriter plugin.

- **ExternalInputExample** – Demonstrates how to directly source data from a Kafka topic (defined in the schema registry with the message format set to Avro).

- **ExternalInputJsonParserExample** – Demonstrates how to directly use a parsed input Json data.

- **ExternalInputXMLParserExample** - Demonstrates how to directly use a parsed input XML data.

- **FilterExample** – Demonstrates how to filter a data source based on a predicate.

- **InfluxDBExample** - Allows periodical dumping of records from a Kafka topic into an InfluxDB output connector. Requires the InfluxDBWriter plugin.

- **JDBCExample** – Allows periodical dumping of records from a Kafka topic into a JDBC database output connector. Requires the JDBCWriter plugin.

- **JoinExample** – Demonstrates how to join a stream to a global table.

- **KdbExample** - Allows periodical dumping of records from a Kafka topic into a Kx kdb+ output connector. Requires the KdbWriter plugin.
MetronomeExample – Demonstrates how the metronome operator works in generating a timestamp field schema. A static metronome has a defined frequency while a dynamic metronome takes frequency as an input which determines the speed of the simulation.

RetentionTimeExample – Demonstrates how to define the different retention time periods set for tables, input streams, output streams, and topics in an application. This helps minimize memory utilization and the amount of data retrieved when subscribing from the beginning to the latest messages.

Note: Setting these properties in the application level overrides the defaults set in the streams.properties file.

For example, if the following properties are defined in the streams.properties file:

```
cep.kafka.table.retention.ms=86400000
cep.kafka.input.retention.ms=60000
cep.kafka.output.retention.ms=900000
```

In the application level, the input retention period will be 1,000 milliseconds instead of 60,000 and the output retention period will be 1,000 milliseconds instead of 900,000. Also, a custom topic retention period has been added using the following pattern: TopicName.retention.ms (i.e., TimeSeries.retention.ms).

```
<properties>
  <!-- Keep tables alive one day -->
  <entry>
    <key>table.retention.ms</key>
    <value>86400000</value>
  </entry>
  <!-- Keep input and output streams for 1 second -->
  <entry>
    <key>input.retention.ms</key>
    <value>1000</value>
  </entry>
  <entry>
    <key>output.retention.ms</key>
    <value>1000</value>
  </entry>
  <!-- Custom retention time for InputStream topic -->
  <entry>
    <key>TimeSeries.retention.ms</key>
    <value>1111</value>
  </entry>
</properties>
```

StockMarketSimulator – Shows a stock market simulation using a streaming data with join, calculations, and metronome operators.

StockStaticTimeSeriesApp – Joins a static and a time series data sources using common keys. Also demonstrates adding a sum aggregation.

StreamtoGlobalTableJoinExample – Joins stream and global table inputs using common keys.

StreamToTableJoinExample - Joins stream and table inputs using common keys.

TextExample - Allows periodical dumping of records from a stream Kafka topic into a Text connector. Requires the TextWriter plugin.

UnionExample- Unioning of two streams.

WindowedStreamExample – Demonstrates aggregation across a windowed stream.
[Appendix]

Properties: Streams

The streams.properties file located in the App_Data folder (i.e., c:\streamsserverdata), contains majority of properties for controlling the configuration of the Panopticon Streams. The following properties can be overridden by updating the file.

<table>
<thead>
<tr>
<th>Property</th>
<th>Authentication: Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>authentication.header.role.delimiter</td>
</tr>
<tr>
<td>Description</td>
<td>The delimiter used to separate the roles. Example: role1, role2,role3</td>
</tr>
<tr>
<td>Default Value</td>
<td>, (Comma)</td>
</tr>
<tr>
<td>Property</td>
<td>Authentication: Header</td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.header.roles</td>
</tr>
<tr>
<td>Description</td>
<td>The name of the header that contains all the roles.</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Authentication: Header</td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.header.username</td>
</tr>
<tr>
<td>Description</td>
<td>The name of the header that contains the username</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Authentication: OAuth 2.0</td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.oauth2.client.ID</td>
</tr>
<tr>
<td>Description</td>
<td>The ID of the OAuth 2.0 client.</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Authentication: OAuth 2.0</td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.oauth2.client.secret</td>
</tr>
<tr>
<td>Description</td>
<td>The secret used by the OAuth 2.0 client.</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Authentication: OAuth 2.0</td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.oauth2.identity.attribute.username</td>
</tr>
<tr>
<td>Description</td>
<td>The attribute that will be extracted from the identity response and used as the username.</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>authentication.oauth2.identity.url</td>
<td>The URL to the REST service that provides details about the authenticated user.</td>
</tr>
<tr>
<td>authentication.oauth2.login.callback.url</td>
<td>The callback URL. The URL should be the same as one of the specified callback URLs used by the client. The URL should refer to the Panopticon Streams</td>
</tr>
<tr>
<td>authentication.oauth2.login.response.type</td>
<td>The response type. The only response type that is currently supported is CODE. The value can also be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.scope</td>
<td>The requested scope. The field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.state</td>
<td>The requested state. The field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.login.url</td>
<td>The URL to the OAuth 2.0 login resource.</td>
</tr>
<tr>
<td>authentication.oauth2.logout.url</td>
<td>The URL to the OAuth 2.0 logout resource. This field can be left blank.</td>
</tr>
<tr>
<td>authentication.oauth2.redirect</td>
<td>Redirects the user back to the Panopticon Streams URL. This is mainly used with a proxy. In which case, the Panopticon Streams does not know the endpoint which the user is going towards to, and therefore cannot redirect the user back to the Overview page. This can be left blank.</td>
</tr>
</tbody>
</table>

Property: Authentication: OAuth 2.0
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>authentication.oauth2.token.method</code></td>
<td>The method on how the token should be retrieved. Supported values are <strong>QUERY</strong>, <strong>BODY</strong>, and <strong>HEADER</strong>.</td>
<td></td>
</tr>
<tr>
<td><code>authentication.oauth2.token.url</code></td>
<td>The URL to the OAuth 2.0 token resource.</td>
<td></td>
</tr>
<tr>
<td><code>authentication.required</code></td>
<td>This property will make the authentication required. It will force the user to login in order to use any of the services provided by the server.</td>
<td><strong>false</strong></td>
</tr>
<tr>
<td><code>authentication.role</code></td>
<td>The required role or group that the user needs to be identified as a Panopticon user. The property can be left blank if no role or group is required.</td>
<td><strong>User</strong></td>
</tr>
<tr>
<td><code>authentication.saml.assertion.roles</code></td>
<td>User attribute for roles configured in the IdP.</td>
<td></td>
</tr>
<tr>
<td><code>authentication.saml.assertion.username</code></td>
<td>User attribute for username configured in the IdP.</td>
<td></td>
</tr>
<tr>
<td><code>authentication.saml.assertionconsumerservice.url</code></td>
<td>The URL to the Panopticon assertion consumer service. URL: <code>[Protocol]://[Host]:[Port]/[Context]/server/rest/auth/login</code> Example: <code>http://localhost:8080/panopticon/server/rest/auth/login</code></td>
<td></td>
</tr>
<tr>
<td><code>authentication.saml.callback.url</code></td>
<td>Relay state.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Default Value</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>---------------</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.certificate.name</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The name of the certificate used to validate signature.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.certificate.password</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The password of the certificate used to validate signature.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.identityprovider.logout.url</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The URL to the IdP logout service.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.identityprovider.url</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The URL to the IdP login service.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.keystore.file</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The location of the Keystore file that contains the certificate.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.keystore.password</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The password to the Keystore file.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.redirect</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Redirects the user back to the Panopticon Streams URL. This is mainly used with a proxy. In which case, the Panopticon Streams does not know the endpoint which the user is going towards to, and therefore cannot redirect the user back to the Overview page. This can be left blank.</td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td><strong>Authentication: SAML</strong></td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>authentication.saml.serviceprovider.ID</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>The ID of the service provider configured in the IdP.</td>
<td></td>
</tr>
</tbody>
</table>

Reference Guide 17.4.0  203
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>authentication.timeout.callback</td>
<td>The timeout (in milliseconds) for the user between initiated login and callback. The default value is five minutes.</td>
<td>300000</td>
</tr>
<tr>
<td>authentication.token.cookie</td>
<td>Used when sticky load balancer is using cookies.</td>
<td>stoken</td>
</tr>
<tr>
<td>authentication.token.domain</td>
<td>The domain in which the token cookie should be registered under.</td>
<td></td>
</tr>
<tr>
<td>authentication.token.persistence</td>
<td>This property is used to determine if the token should persist if the browser is closed or if it should only last while the browser is open. There are two possible values: PERSISTENT and SESSION. PERSISTENT will persist the token in the browser even if the browser has been closed and reopened. SESSION will remove the token from the browser if it is shutdown.</td>
<td>PERSISTENT</td>
</tr>
<tr>
<td>authentication.token.refreshable</td>
<td>This property determines if the token can refresh itself. The web client can identify if the token is about to expire and then request a new token with the existing token. A token is refreshable if the property is set to true. The token will expire and invalidate the user session if the property is set to false.</td>
<td>true</td>
</tr>
<tr>
<td>authentication.token.refreshable.scope</td>
<td>This property determines who can refresh a token: ALL or CLIENT. ALL means that both the client and the internal subscriptions can refresh a token for a user. This ensures that an internal subscription will always be authenticated. CLIENT means that only the client can refresh the token. This prohibits the server from refreshing a token for an internal subscription. Therefore, it is beneficial in ensuring a user interaction is required to refresh the token. This is recommended when there are more security constraints and a short lifespan on the token.</td>
<td>ALL</td>
</tr>
<tr>
<td>authentication.token.secret</td>
<td>The secret is used to sign the token. The secret will be auto-generated when the server starts for the first time.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Default Value</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Authentication</td>
<td>The number of seconds that the token should be valid.</td>
<td>604800</td>
</tr>
<tr>
<td>Authentication</td>
<td>The type of the authentication mechanism that will be used on the Panopticon Streams.</td>
<td>BASIC</td>
</tr>
<tr>
<td>Cache</td>
<td>The ID of the cache plugin that will be used. Possible value: BinaryTableFile-Cache</td>
<td>BinaryTableFile-Cache</td>
</tr>
<tr>
<td></td>
<td>The condition for determining when the cache should be purged or cleared. Possible values: NONE, MEMORY.</td>
<td>MEMORY</td>
</tr>
<tr>
<td></td>
<td>The memory threshold used to determine if the cache should be purged or not. The values are presented in percent, 0-100. 80 means that the cache will be purged if the memory consumption reaches 80 % or more.</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>Enable or disable the purge functionality. Possible values: true, false</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>The rate on how often the purge condition should be checked. The value is presented in milliseconds.</td>
<td>10000</td>
</tr>
</tbody>
</table>

NOTE: This value should be kept a secret.
<table>
<thead>
<tr>
<th>Description</th>
<th>Enable the cache clearing schedule. This is scheduling the clear cache operation which will remove all the expired cache entries.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Value</td>
<td><strong>true</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.schedule.clear.rate</code></td>
</tr>
<tr>
<td>Description</td>
<td>Set an interval for how often the clear cache scheduler will clear the expired cache entries</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>600000</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.enabled</code></td>
</tr>
<tr>
<td>Description</td>
<td>Enables and disable the service cache</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>true</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.getdata.queue.size</code></td>
</tr>
<tr>
<td>Description</td>
<td>The max size for the cache queue for the service GetData</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>100</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.getdatatabledata.queue.size</code></td>
</tr>
<tr>
<td>Description</td>
<td>The max size for the cache queue for the service GetDatatableData</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>10</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.getformatteddata.queue.size</code></td>
</tr>
<tr>
<td>Description</td>
<td>The max size for the cache queue for the service GetFormattedData</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>10</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.getresourcedworkbook.queue.size</code></td>
</tr>
<tr>
<td>Description</td>
<td>The size on how many workbooks will be cached.</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>100</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.type</code></td>
</tr>
<tr>
<td>Description</td>
<td>The service cache mechanism being used. Allowed values: MEMORY</td>
</tr>
<tr>
<td>Default Value</td>
<td><strong>IN_MEMORY</strong></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>Cache</td>
</tr>
<tr>
<td>Attribute</td>
<td><code>cache.service.usecacheonfailure</code></td>
</tr>
<tr>
<td>Description</td>
<td>Enables the cache fallback functionality on failure. This means that the server will use expired cache even on failure.</td>
</tr>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
<tr>
<td>---------------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Application</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.application.autostart</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Determines whether all of the stored applications in the Streams server should auto start when the Streams server starts.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td>false</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Application</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/application.state.path</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Where the tmp folder of the Panopticon Streams data are created.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td>C:/PanopticonStreams/Data/tmp/kafka-streams</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/connection.timeout</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The connection timeout towards Kafka. The value is presented in milliseconds.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td>10000</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/input.retention.ms</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Specifies the retention period of input streams.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td>60000</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/output.retention.ms</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Specifies the retention period of output streams.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td>900000</td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/path</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The file path to our distributed Kafka, ZooKeeper, Schema registry package. This property is required to be able to start Kafka, ZooKeeper and Schema Registry from the user interface. If this value is not defined, these instances will not be started and Panopticon Streams will simply try to connect to already running ones.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/schemaregistry.url</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The URL to the Schema Registry.</td>
</tr>
<tr>
<td><strong>Default Value</strong></td>
<td><a href="http://localhost:8081">http://localhost:8081</a></td>
</tr>
<tr>
<td><strong>Property</strong></td>
<td>CEP: Kafka</td>
</tr>
<tr>
<td><strong>Attribute</strong></td>
<td>cep.kafka/servers</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The URL to all the Kafka servers.</td>
</tr>
<tr>
<td>Property</td>
<td>Attribute</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>cep.kafka.session.timeout</td>
</tr>
<tr>
<td></td>
<td>cep.kafka.table.retention.ms</td>
</tr>
<tr>
<td></td>
<td>cep.kafka.zookeeper.servers</td>
</tr>
<tr>
<td></td>
<td>cep.type</td>
</tr>
<tr>
<td></td>
<td>documentation.enabled</td>
</tr>
<tr>
<td></td>
<td>error.default.message</td>
</tr>
<tr>
<td></td>
<td>logger.level.file</td>
</tr>
<tr>
<td></td>
<td>rest.enabled</td>
</tr>
</tbody>
</table>

**Default Values**

- **localhost:9092**
- **15000**
- **86400000**
- **localhost:2181**
- **KAFKA**
- **false**
- **WARNING**
- **true**
<table>
<thead>
<tr>
<th>Property</th>
<th>REST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>rest.response.error.stacktrace.included</td>
</tr>
<tr>
<td>Description</td>
<td>Include the error stacktrace in REST responses.</td>
</tr>
<tr>
<td>Default Value</td>
<td>false</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>SOAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>soap.enabled</td>
</tr>
<tr>
<td>Description</td>
<td>Enable or disable the SOAP interface</td>
</tr>
<tr>
<td>Default Value</td>
<td>True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>license.hwu.operating.system</td>
</tr>
<tr>
<td>Description</td>
<td>The operating system where the Panopticon Streams is installed. Possible values are: WIN_X86, WIN_X64, MAC, LINUX_X64, or LINUX_ARM64. <strong>NOTE:</strong> If the Java bitness (e.g., 32-bit) is different from the operating system (e.g., 64-bit), it is recommended to add the Java bitness in this property (e.g., WIN_X86).</td>
</tr>
<tr>
<td>Default Value</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>license.hwu.uri</td>
</tr>
</tbody>
</table>
| Description | The path where the License Server is running e.g., 6200@191.255.255.0 where the syntax is PORTNUMBER@HOST. If multiple servers are used, they should be separated by ‘;’. **NOTES:**  
- Multiple License Servers are not supported when the Panopticon Stream is on a Linux machine.  
- If value is not set in the panopticon.properties, the environment variable ALTAIR_LICENSE_PATH serves as the backup path and will be used |
| Default Value | |

<table>
<thead>
<tr>
<th>Property</th>
<th>Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>license.hwu.version</td>
</tr>
<tr>
<td>Description</td>
<td>Value must match the license version found in the HyperWorks Units license file.</td>
</tr>
<tr>
<td>Default Value</td>
<td>19.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Licensing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>license.mode</td>
</tr>
<tr>
<td>Description</td>
<td>The license mode. Possible values are: FILE or HWU. To use the HyperWorks Units license, set this property to HWU.</td>
</tr>
<tr>
<td>Default Value</td>
<td>FILE</td>
</tr>
</tbody>
</table>