SYBASE®
An SAP® Company

Introduction to Data Modeling and the Aleri Studio

Sybase Aleri Streaming Platform 3.2

DOCUMENT ID: DC01365-01-0320-01

LAST REVISED: December, 2010

countries.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Group Ltd.

All other company and product names mentioned may be trademarks of the respective companies with which they are associated.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Table of Contents

About This Guide

1. Purpose

This tutorial is intended to help users build a few sample data models with the Aleri Studio. It does not provide instructions on how to configure the system to accept live data from a market feed or other external source. Consult the *Authoring Guide* for more detailed information on configurations.

2. Organization

This guide consists of the following chapters:

Chapter 1, *Introduction to the Tutorial*	gives an overview of the Sybase® Aleri Streaming Platform and data models.
Chapter 2, *Data Absorption*	provides instructions for using the Aleri Studio to load, examine and test a simple data model that absorbs data from an external source.
Chapter 3, *Authoring a Filter Stream*	features step-by-step directions for authoring a Filter Stream in a data model using the Aleri Studio.
Chapter 4, *Filtering Streaming Data*	provides instructions for testing the effect of the Filter Stream in a data model.
Chapter 5, *Authoring an Aggregate Stream*	describes how to develop the data model still further by authoring an Aggregate Stream.
Chapter 6, *Aggregating Streaming Data*	gives instructions for viewing the Aggregate Stream's action on uploaded trade data.
Chapter 7, *Joining Data from Two Streams*	elaborates on directions for loading, examining and testing a more complex version of the data model in the Aleri Studio.

Note:

The “Authoring” chapters above are for people who want hands-on practice with authoring new streams in a data model using Aleri Studio tools. This tutorial was designed so that these chapters can be skipped if you just want to examine and test data models that have already been prepared and installed with the Sybase Aleri Streaming Platform.

3. Related Documents

This guide is part of a set. The following list briefly describes each document in the set.

Product Overview	Introduces the Aleri Streaming Platform and related Aleri products.
Getting Started - the Aleri Studio	Provides the necessary information to start using the Aleri Studio for defining data models.
Release Bulletin	Describes the features, known issues and limitations of the latest Aleri Streaming Platform release.
Installation Guide	Provides instructions for installing and configuring the Streaming

	Processor and Aleri Studio, which collectively are called the Aleri Streaming Platform.
Authoring Guide	Provides detailed information about creating a data model in the Aleri Studio. Since this is a comprehensive guide, you should read the *Introduction to Data Modeling and the Aleri Studio* first.
Authoring Reference	Provides detailed information about creating a data model for the Aleri Streaming Platform.
Guide to Programming Interfaces	Provides instructions and reference information for developers who want to use Aleri programming interfaces to create their own applications to work with the Aleri Streaming Platform. These interfaces include: • the Publish/Subscribe (Pub/Sub) Application Programming Interface (API) for Java • the Pub/Sub API for C++ • the Pub/Sub API for .NET • a proprietary Command & Control interface • an on-demand SQL query interface
Utilities Guide	Collects usage information (similar to UNIX® man pages) for all Aleri Streaming Platform command line tools.
Administrators Guide	Provides instructions for specific administrative tasks related to the Aleri Streaming Platform.
Introduction to Data Modeling and the Aleri Studio	Walks you through the process of building and testing an Aleri data model using the Aleri Studio.
SPLASH Tutorial	Introduces the SPLASH programming language and illustrates its capabilities through a series of examples.
Frequently Asked Questions	Answers some frequently asked questions about the Aleri Streaming Platform.

Chapter 1. Introduction to the Tutorial

This tutorial shows how to use the Aleri Studio, to build and run the data model, and see the results. It isn't the only way to build a data model; you can also use Aleri SQL or AleriML and a text editor. But the Aleri Studio, is the easiest way for newcomers to see how the Sybase Aleri Streaming Platform works.

The Sybase Aleri Streaming Platform runs a “data model.” The basic building block of a data model is a “stream.” There are two types of streams in a data model: source streams and derived streams. A source stream is an entry point for incoming data. A derived stream is the result of operations being applied to one or more other streams in the model.

This tutorial will show how to build a simple model that continuously updates the value of a stock portfolio as incoming market prices are received. The simple model consists of six streams: three that receive data, two that perform intermediate computations, and one that produces the final result. But a model for production can contain any number of source streams and derived streams. A model is also not limited to a single result stream so that any stream in the model can publish its results as a real-time output stream. Additionally, you can view a stream's data “on demand” by submitting an ad-hoc query.

The tutorial starts with a source stream that receives an incoming market data feed for stock prices. To simulate a price feed, you would run the data model with an “upload” utility that reads data from a file and streams it into the Sybase Aleri Streaming Platform. The data file contains 1,000 records, each reporting a trade in the stock market and the price of the trade.

The rest of the streams in the model will be added after this point. The ultimate result will be in a stream that shows the value of the portfolio and updates it in real time.

The files used in this tutorial are put in the `examples/tutorial` subdirectory, of the installation. If you have not already done so, install the software before proceeding with the tutorial. Refer to the *Installation Guide* for instructions.

Chapter 2. Data Absorption

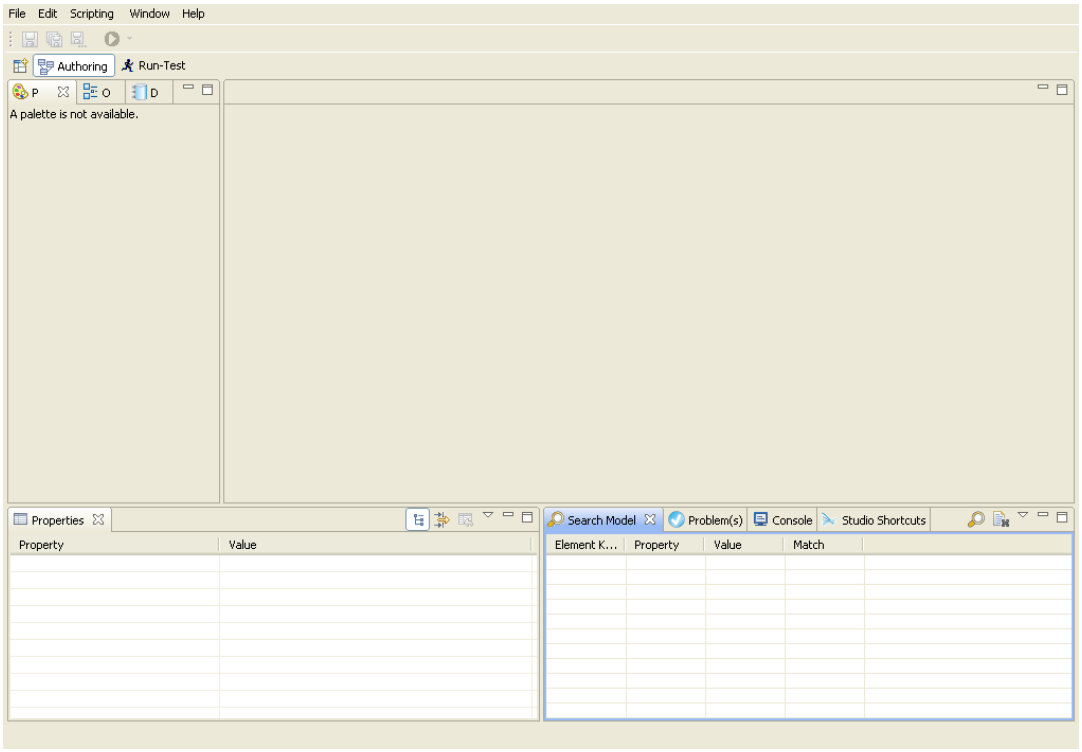
This tutorial is based on a series of data models, which are to be loaded into the Aleri Studio as directed. Each model builds on the previous one. The goal of the tutorial is to show how a simple model can be developed to do more sophisticated processing.

The most basic data model is a single source stream — a stream with a function to absorb data into the Sybase Aleri Streaming Platform from an external source. For this tutorial, the data source is a file containing records of 1000 trades. The upload utility will move the data to the Sybase Aleri Streaming Platform and watch as the Source Stream receives it.

Load the first model

1. Start the Aleri Studio.
2. When the Aleri Studio Welcome Page is displayed, click on the **AleriStudio** button in the lower right corner.

3. When the Aleri Studio Main Menu is displayed:

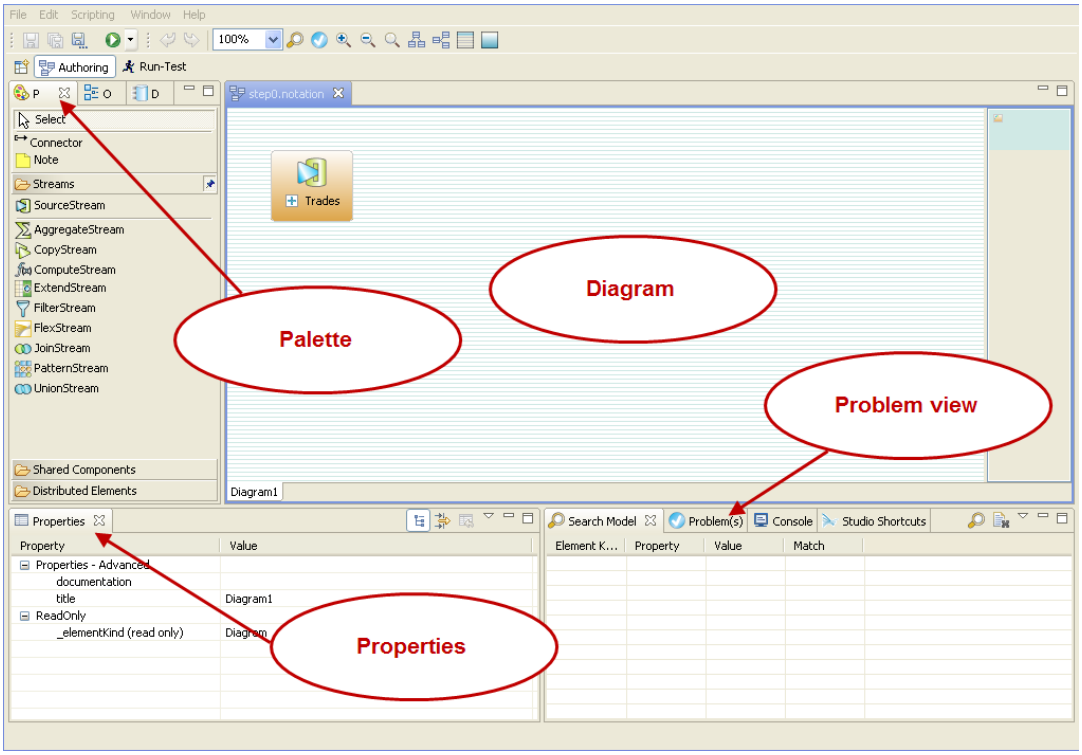


4. Select **Open** from the **File** menu.

   In the **Open** dialog box, browse to the folder containing the appropriate `.notation` file for this step in the tutorial. A notation file is a data model in Aleri Studio format. By convention, these file names have `.notation` at the end. You will find subdirectories containing the appropriate `.notation` file for each step of this tutorial in the directory `PLATFORM_HOME/examples/tutorial` (where *`PLATFORM_HOME`* is the directory in which the Sybase Aleri Streaming Platform has been installed). On a Windows® PC, this is typically the `C:\My Documents\Aleri` folder. On Linux® and Solaris® systems, this is the `/home/aleri/aleri/platform/3.2.0/examples` directory unless you specified an alternate location when you installed the software.

   Select `step0.notation` and click on **Open**.

   The `step0` data model is loaded into the Aleri Studio, and displayed in the Aleri Studio Authoring perspective, which is one of the two perspective tabs shown (the other is the Run-Test perspective).



5. Look over the main sections of the Aleri Studio Authoring perspective:

Diagram window	Displays the model's components in graphical form. This model consists of one source stream, named *Trades*.
Palette	Displays the items that can be created by dragging them on to the diagram.
Properties tab panel	Displays the basic properties of the selected stream.
Problem view	Displays any errors found in the model, when checked.

6. Click on the **Trades** shape in the diagram so that information about the Trades stream appears in the **Properties** tab panel.

   The Aleri Studio allows the creation of new streams to run on the Sybase Aleri Streaming Platform. You can specify each stream's properties and define the dataflow between streams.

   The Trades stream, Aleri Studio's Properties tab panel, allows you to view and/or edit that stream's basic properties.

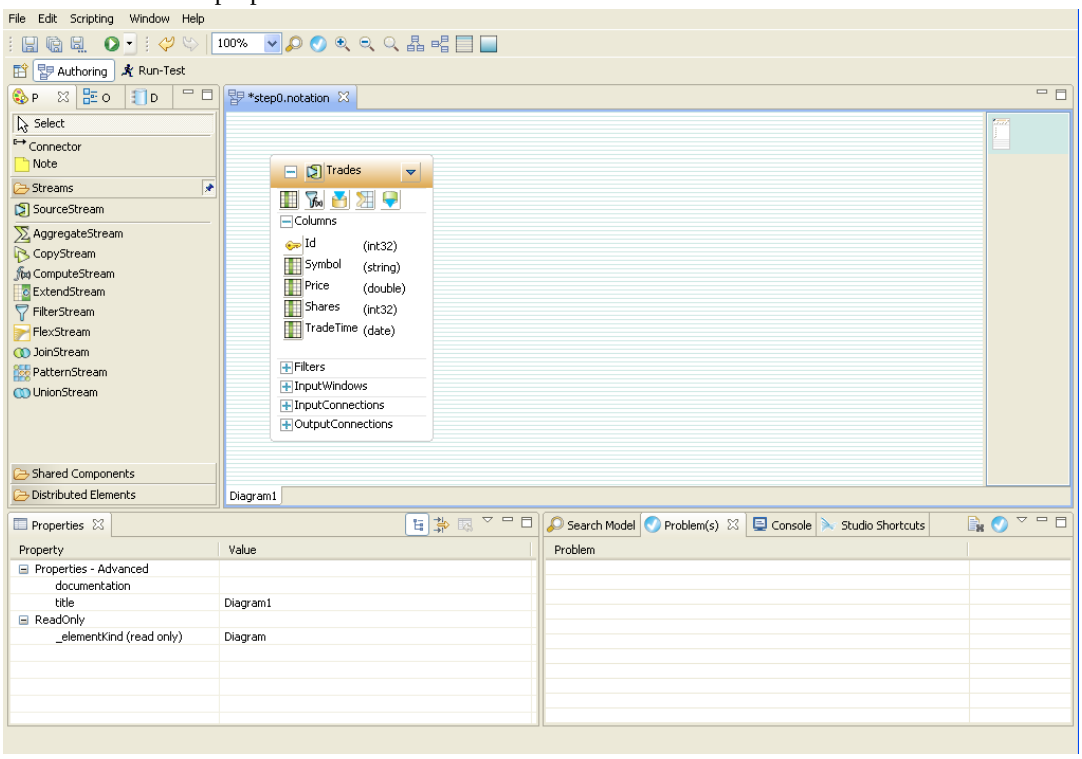
   An Sybase Aleri Streaming Platform stream is very similar in structure to a table in a relational database (although the two are very different in action). A stream works with rows and columns similar to a table or materialized view. Each row stores a new data record; the columns defined for the stream specify what types of data are stored in each field of the row.

   The following columns are defined for the Trades stream:

Column Name	**Datatype**
Id	int32

Symbol	string
Price	double
Shares	int32
TradeTime	date

7. Click on the + on the **Trades** shape in the diagram so that the shape is replaced by a box showing the Trades stream's properties.

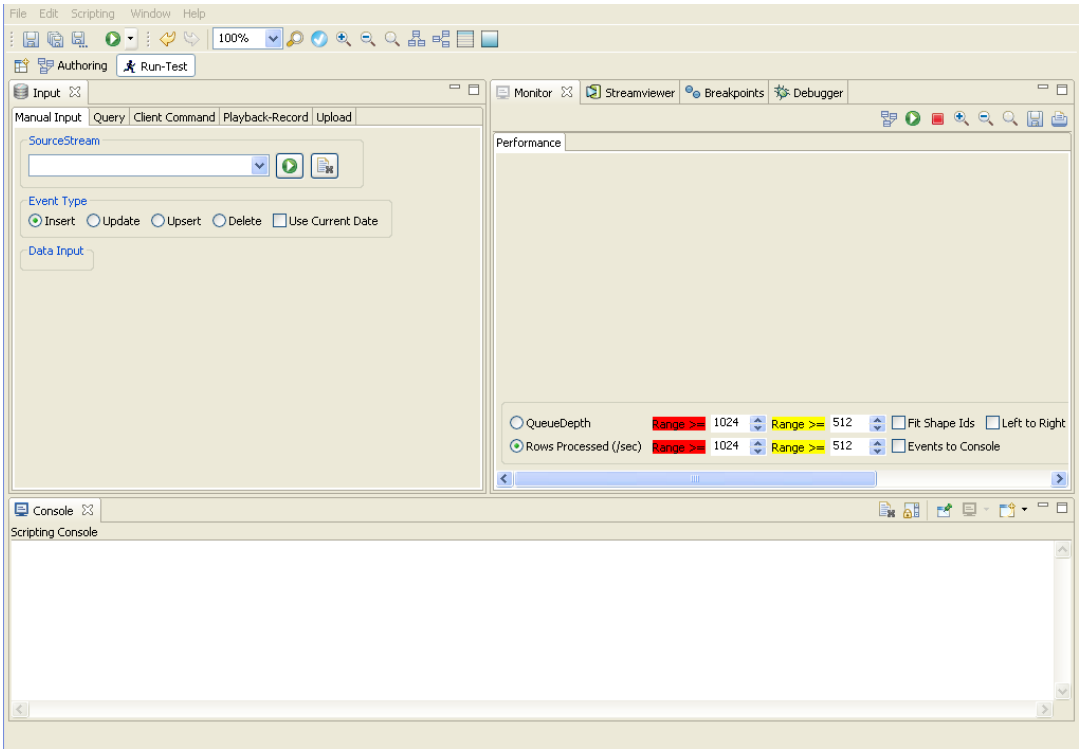


The + and - on the stream toggles (switching back and forth between two values) the image between the “Image ” and the “Compartment modes”. The stream shape in the Compartment mode shows the stream's basic properties, including the columns defined for it. The Compartment mode gives you a quick way to view and/or edit these properties, but streams represented in this mode take up a lot of room in the diagram. All the visible properties in the Compartment mode can also be edited in the **Properties** tab panel, no matter which mode the shape appears in.

Click the - to switch the shape back to the Image mode.

Configure and Start the Sybase Aleri Streaming Platform

8. Click the **Run-Test** tab to display the Run-Test perspective.



The Aleri Studio is not just an authoring environment; it also provides a test environment. The Run-Test perspective allows you to push data to the Sybase Aleri Streaming Platform, view processed data, see performance statistics, and debug data models.

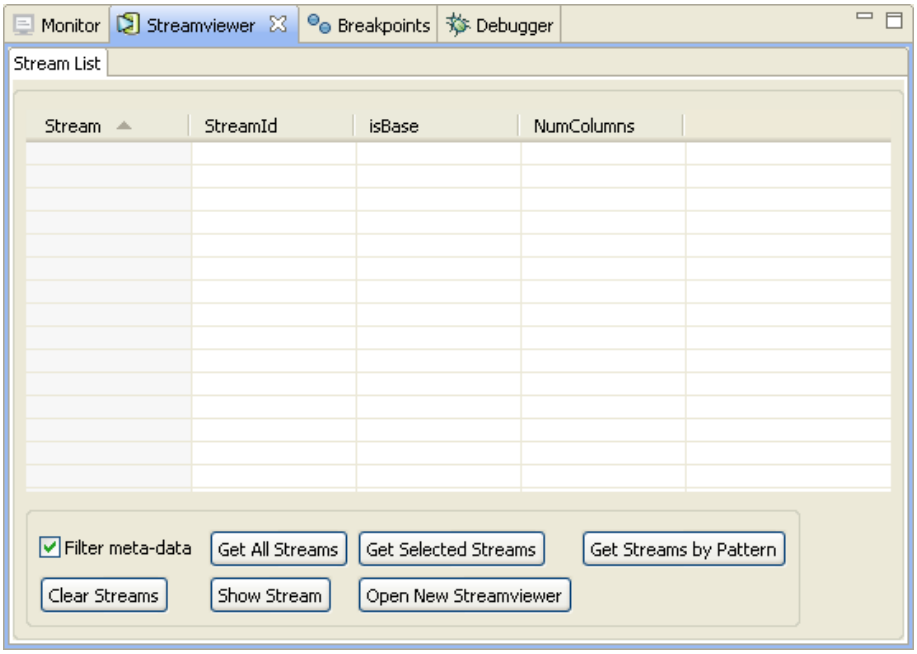
For now, you will most likely be running the Sybase Aleri Streaming Platform on the same computer as the Aleri Studio (this configuration is referred to as local execution). In a production environment, the two components would likely run on separate computers (a configuration referred to as remote execution). The Run-Test perspective allows you to configure and run the Sybase Aleri Streaming Platform even if it is on a different computer or type of hardware.

9. Click the green arrow button under the **Scripting** menu. This will start the Sybase Aleri Streaming Platform with the data model. After a moment, the green arrow button should turn into a red box.

Set up the Streamviewer

Now that the Sybase Aleri Streaming Platform has been started, the Trades stream is ready to absorb incoming data. You can use the Streamviewer facility to create a subscription to the Trades stream and view the stream's contents.

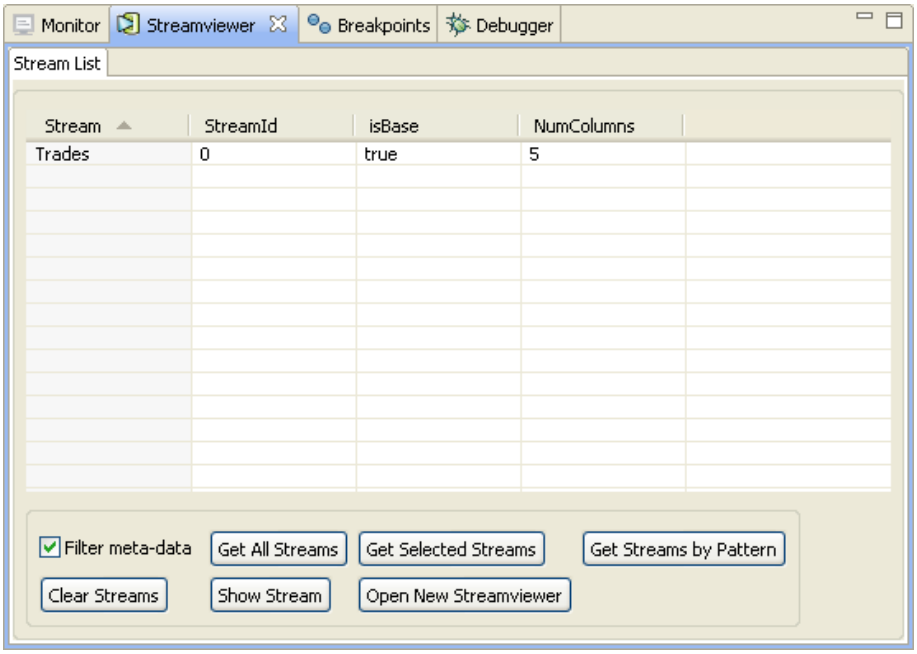
10. On the right side of the **Run-Test** perspective, click the **Streamviewer** tab to bring this tab panel to the front.



11. In the **Streamviewer** tab panel, check the **Filter meta-data** check box in the lower left corner of the **Viewer** sub-panel. Checking this box tells the Streamviewer to list only the streams defined in the data model.

    Click **Get All Streams**.

    A list of the streams in this data model is displayed in the **Viewer** sub-panel. (In this case there is only one stream, Trades.)



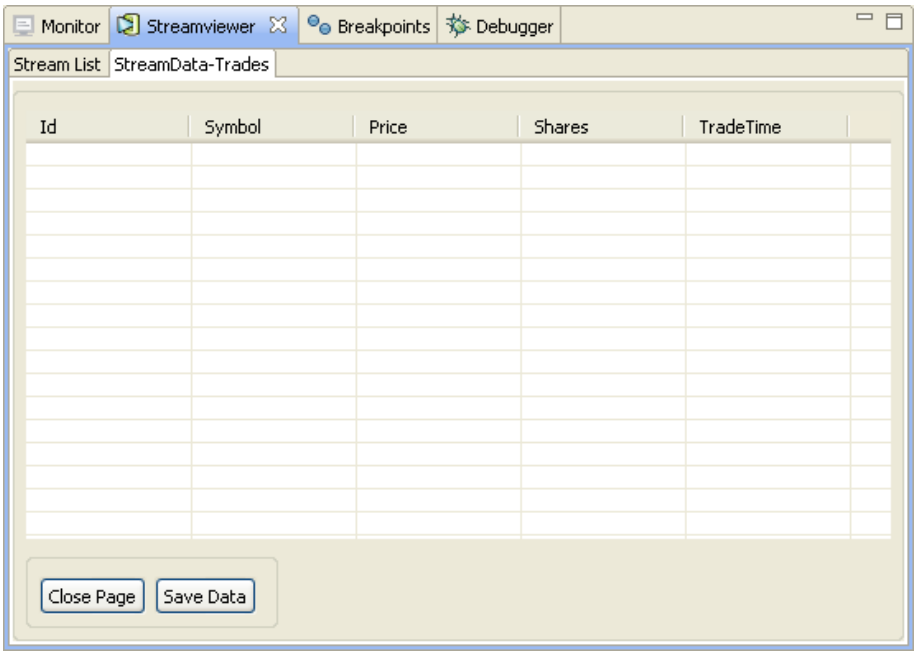
Note:

"Meta-data streams" are streams maintained internally by the Sybase Aleri Streaming Platform.

They hold information about the other streams defined within the data model.

The meta-data contained in these streams can be useful in advanced modeling, but it is not used in this tutorial.

12. Double-click on the **Trades** row. The Aleri Studio will then create a new sub-panel called **Stream-Data-Trades** and bring this sub-panel to the front.



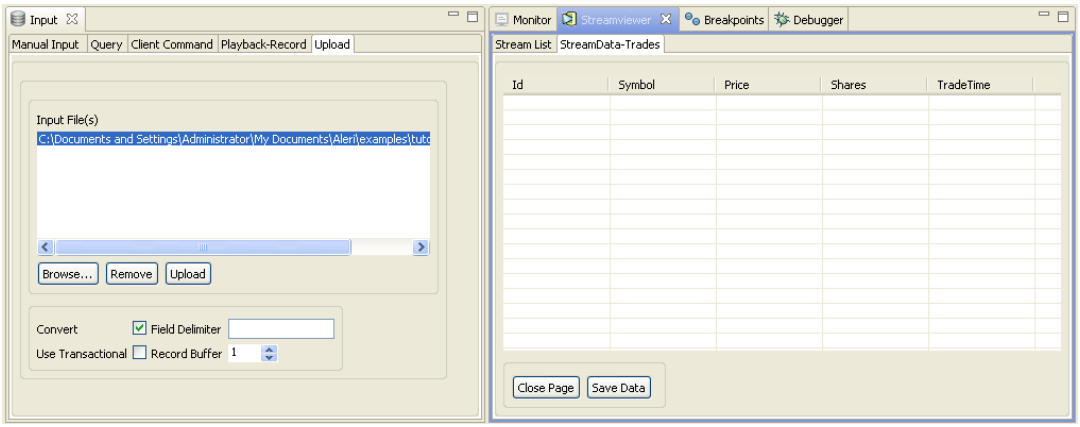
The columns of this tab panel are the same as the columns defined for the Trades stream. This tab panel is a real-time display of the output from the subscription to the Trades stream.

Upload the Trade data

This tutorial provides instructions on building a model that shows the Sybase Aleri Streaming Platform uploading and processing 1000 sample records stored in the file `Trades.xml`, which is provided as part of the Sybase Aleri Streaming Platform installation.

The Aleri Studio includes an **Upload** tab panel that can be used to select a data file to upload into the Sybase Aleri Streaming Platform. This data is received and processed just as if it were coming from a live data feed. This functionality is invaluable for testing and debugging a data model before it goes live.

13. In the **Input** tab panel, click the **Upload** tab to bring this sub-panel to the front. Click the **Browse...** button. In the **Select Input File** dialog box, browse to the file `Trades.xml`, which is in the same directory as the `.notation` file opened in an earlier step of the tutorial. Then click **Open**. The path and filename you chose appears in the **Input File(s)** field in the **Upload** tab panel.

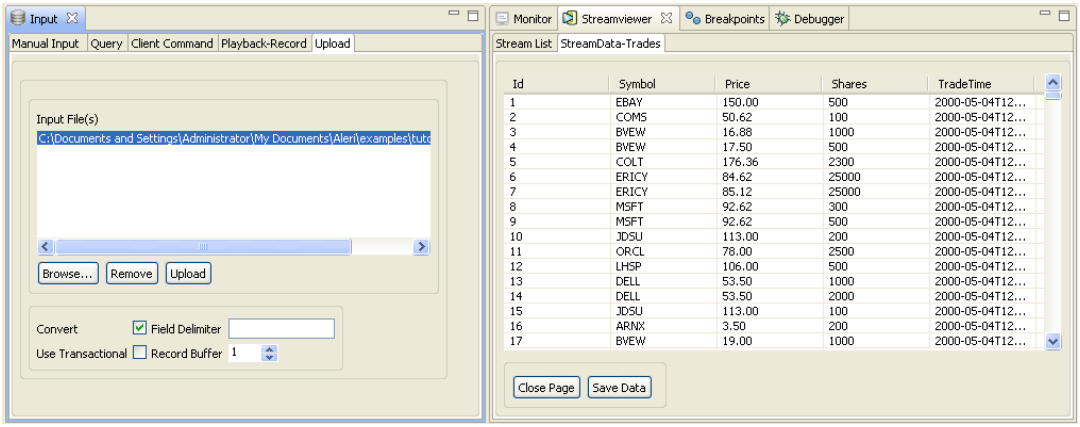


14. Click once on `Trades.xml`, in the **Input File(s)** field to highlight it.

    Check the **Convert** check box in the lower left corner of the **Upload** tab panel (to convert the XML data in the selected file to binary data, which will then be uploaded to the Sybase Aleri Streaming Platform).

    Click **Upload**. The trade records from `Trades.xml` are converted to binary data and uploaded into the Trades stream.

    As this happens, these records are displayed in the **StreamData-Trades** panel of the Streamviewer.



    Messages related to the upload are displayed in the **Aleri Server Console** tab panel.

15. Scroll through the rows in the **StreamData-Trades** sub-panel and note the following:

    - The trades listed have several different Symbol values (*`EBAY`*, *`COMS`*, *`MSFT`*, *`IBM`*, and others). You can sort the display by Symbol by clicking once on the heading of the *`Symbol`* column.
    - The values in the *`TradeTime`* column are not current. These trades are from a file that was generated in the year 2000.
    - Counting by *`Id`* number, you can see that there were 1000 trade records uploaded to the Sybase Aleri Streaming Platform.

16. Click the red stop square under the **Scripting** menu to stop the Sybase Aleri Streaming Platform.

    The Sybase Aleri Streaming Platform is stopped. The rows of data in the **StreamData-Trades** sub-panel disappear (since the stream no longer exists in memory).

Messages sent out from the Sybase Aleri Streaming Platform as it shuts down appear in the **Aleri Server Console** tab panel. If a different console appears, you can click the **Display Selected Console** icon on the Console Toolbar and select **Aleri Server Console**.

17. You are now ready to take go to the next step in this tutorial where you learn how to introduce a new derived stream to filter the Trades data.

    - To author this new stream yourself, go to Chapter 3, *Authoring a Filter Stream*.
    - To skip the process of authoring the new stream by just loading a new version of the data model that has the stream already, go to Chapter 4, *Filtering Streaming Data*.

Chapter 3. Authoring a Filter Stream

The next development of this tutorial's data model is the addition of a Filter Stream.

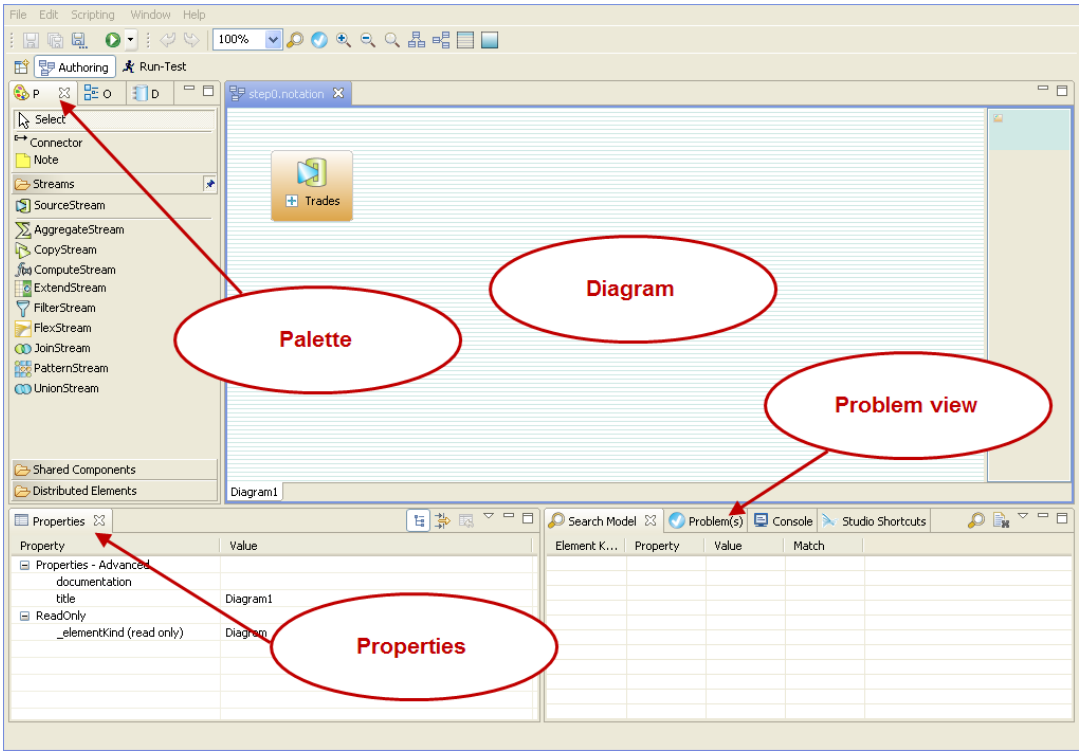
Get started

The first step is to make sure the Aleri Studio is running but not the Sybase Aleri Streaming Platform.

1. Start this part of the tutorial as follows:
   - If the Aleri Studio is already running: make sure the Sybase Aleri Streaming Platform is not running. If it is, click the red stop button to stop it.
   - If the Aleri Studio is not running:
     - Start the Aleri Studio.
     - Click the **AleriStudio** button in the lower right corner of the Welcome Page, and the main Aleri Studio window appears.
2. Select **Open** from the **File** menu.

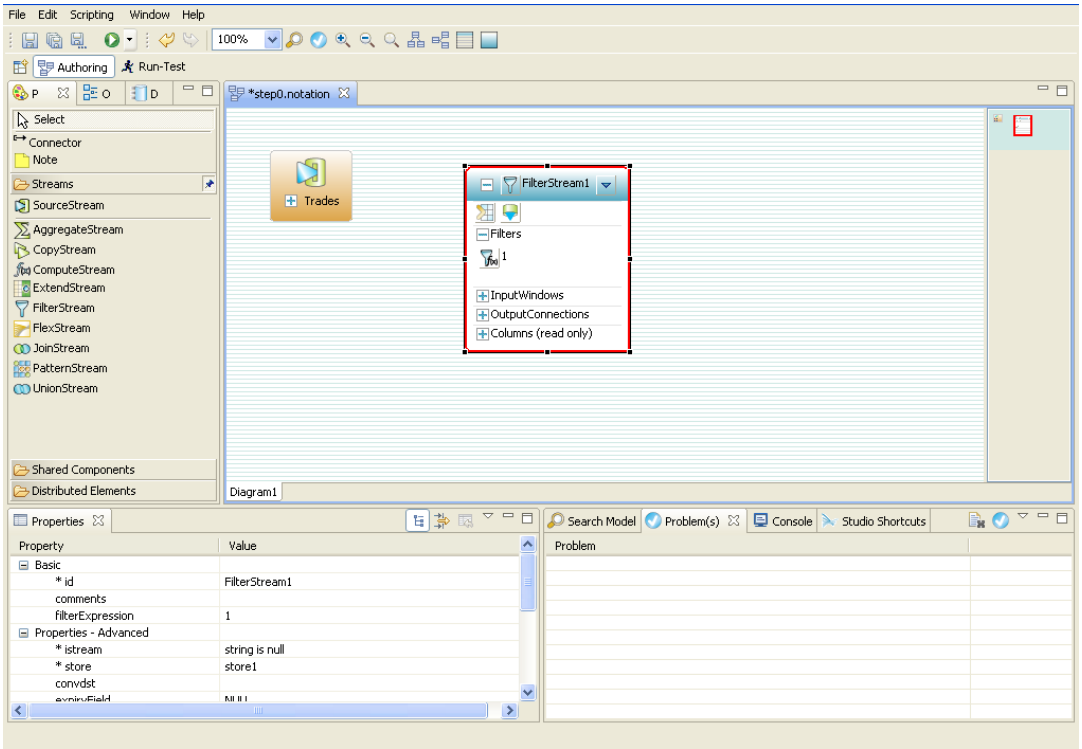
   In the **Open** dialog box, browse to the folder containing the appropriate `.notation` file for this step in the tutorial. You will find subdirectories containing the appropriate `.notation` file for each step of this tutorial in the directory `$PLATFORM_HOME/examples/tutorial` (where *`$PLATFORM_HOME`* is the directory in which the Sybase Aleri Streaming Platform has been installed). On a Windows® PC, this is typically the `My Documents\Aleri` folder. On Linux® and Solaris™ systems, this is the `/home/aleri/aleri/platform/3.2.0/examples` directory unless you specified an alternate location when you installed the software.

   Select `step0.notation` and click **Open**. The `step0` data model is loaded into the Aleri Studio, and displayed in the Authoring perspective.



3. Click **Filter Stream** in the **Palette** tab panel on the left side of the screen.

   Click once in the diagram area (at about the same height at the top of the *Trades* shape). A new Filter Stream appears in the diagram, in the Compartment mode:



The stream shape is outlined in red; it will retain this red outline until it has been connected to an input stream.

4. Click on *`FilterStream1`* in the shape to highlight the text.

5. Type the name `WatchTrades` in this line, and press **Enter** to set the new name.

Connect the Filter Stream to the Source Stream

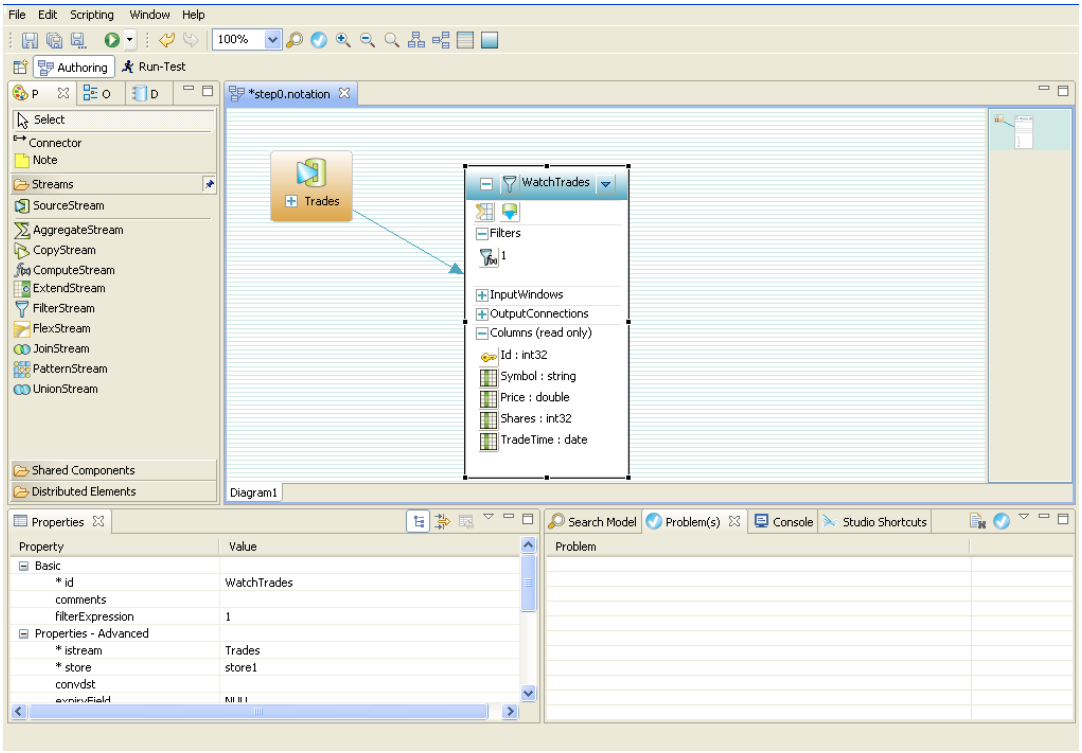
6. Click on **Connector** in the **Palette** tab panel.

   Move the mouse cursor over the diagram area. The cursor arrow now has a “plug” graphic attached to its tail.

7. Click once on the **Trades** shape, then move the cursor over to the new **WatchTrades** shape.

   Click once on the *`WatchTrades`* shape.

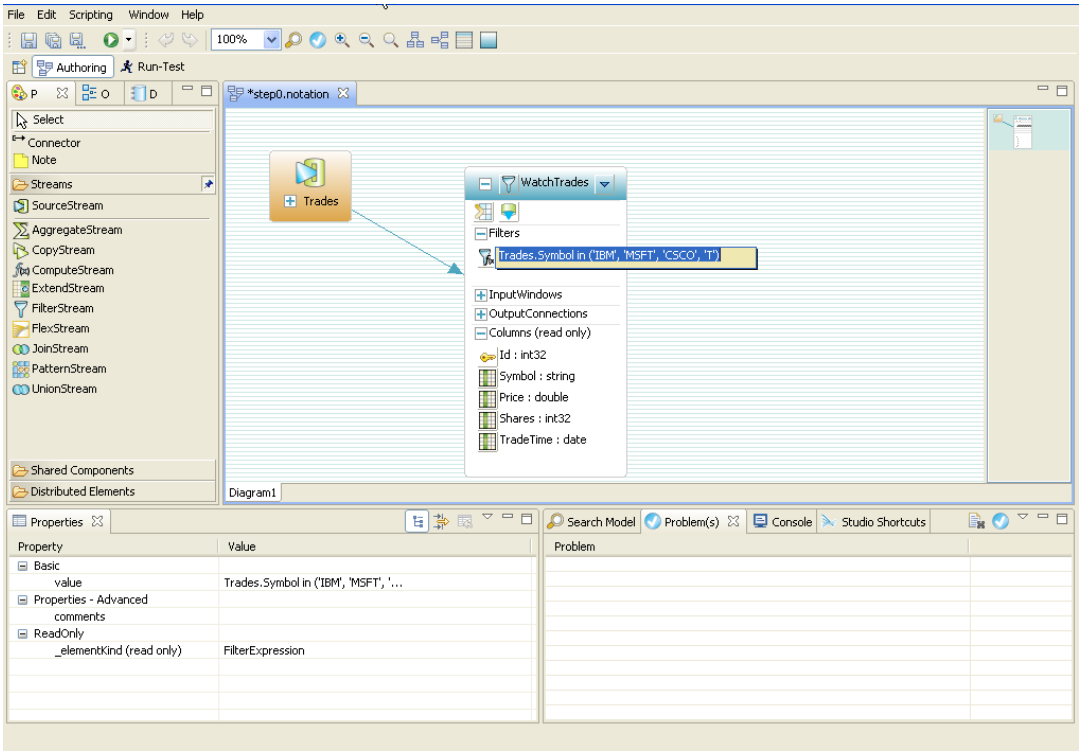
   - An arrow appears from the **Trades** shape to the **WatchTrades** shape.
   - The outline of the **WatchTrades** shape changes to black.



8. The new Filter Stream could be used at this point, but it would not do any filtering. To make it useful, you must add the filter expression that the Filter Stream will apply to each event it receives. See the section called “Load the second data model” for an explanation of how the Filter Stream works.

Enter the FilterExpression

9. Click on *filterExpression* entry in the *WatchTrades* shape, and press **F2** to open the inline editor.



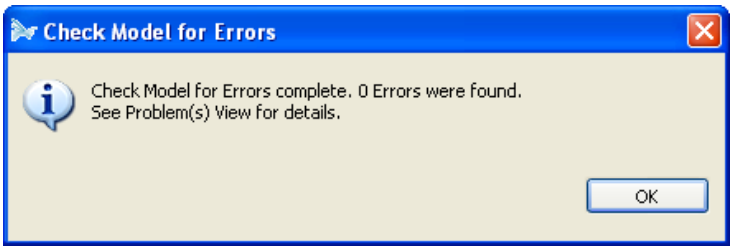
This window displays the current value (`1`) of the *filterExpression* property.

10. Replace `1` with the expression

```
Trades.Symbol in ('IBM', 'MSFT', 'CSCO', 'T')
```

and press **Enter**.

11. Select **Check Syntax** from the top tool bar (it is a blue check mark). The Aleri Studio checks the expression just typed in. If there are no problems with the expression, the following message box appears:



Note:

If the syntax check does report an error, check the spelling of your entry, make the necessary fixes, and try again.

Click **OK** to close the message box.

12. Select **Save As...** from the **File menu**.

Save the revised data model under a new name.

13. Exit the Aleri Studio by selecting **Exit** from the **File menu**. The Aleri Studio closes.

You are now ready to stream data through the Filter Stream and see the results.

Chapter 4. Filtering Streaming Data

The first step of this tutorial demonstrated how a simple Source Stream introduces streaming data into the Sybase Aleri Streaming Platform. This next step shows one simple way in which the Sybase Aleri Streaming Platform can filter the data it receives.

Load the second data model

Start by making sure that the Aleri Studio is running but not the Sybase Aleri Streaming Platform.

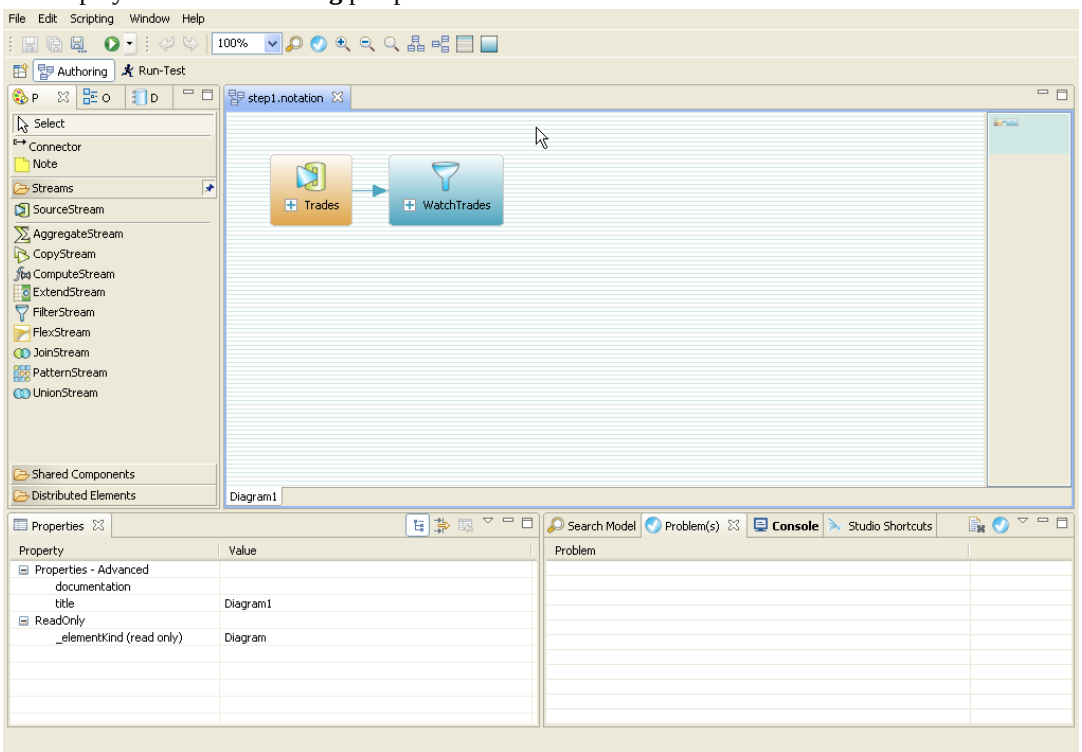
1. Start this part of the tutorial as follows:

   - If the Aleri Studio is already open, make sure the Sybase Aleri Streaming Platform is not running. If it is, click the red stop button to stop it.
   - If the Aleri Studio is not open,
     - Start the Aleri Studio.
     - Click the **AleriStudio** button in the lower right corner of the Welcome Page, and the main Aleri Studio window appears.

2. Select **Open** from the **File** menu.

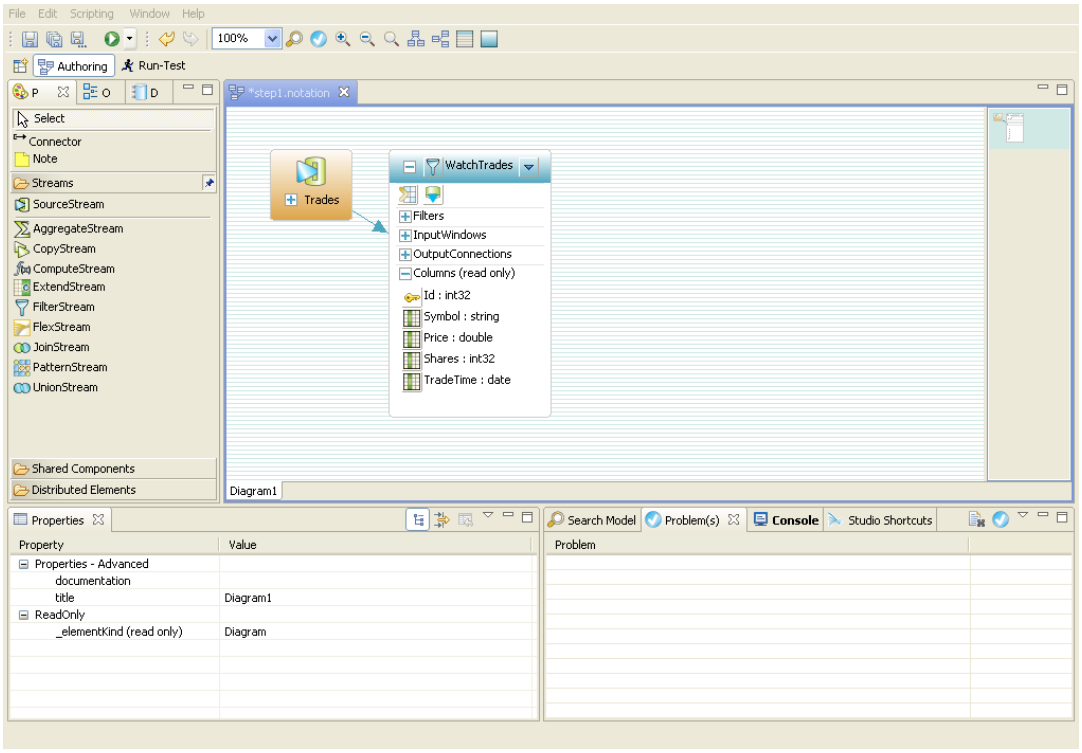
   In the **Open** dialog box, browse to the directory containing the `step1.notation` file for the tutorial.

   Select `step1.notation` and click **Open**. The `step1` data model is loaded into the Aleri Studio, and displayed in the **Authoring** perspective.



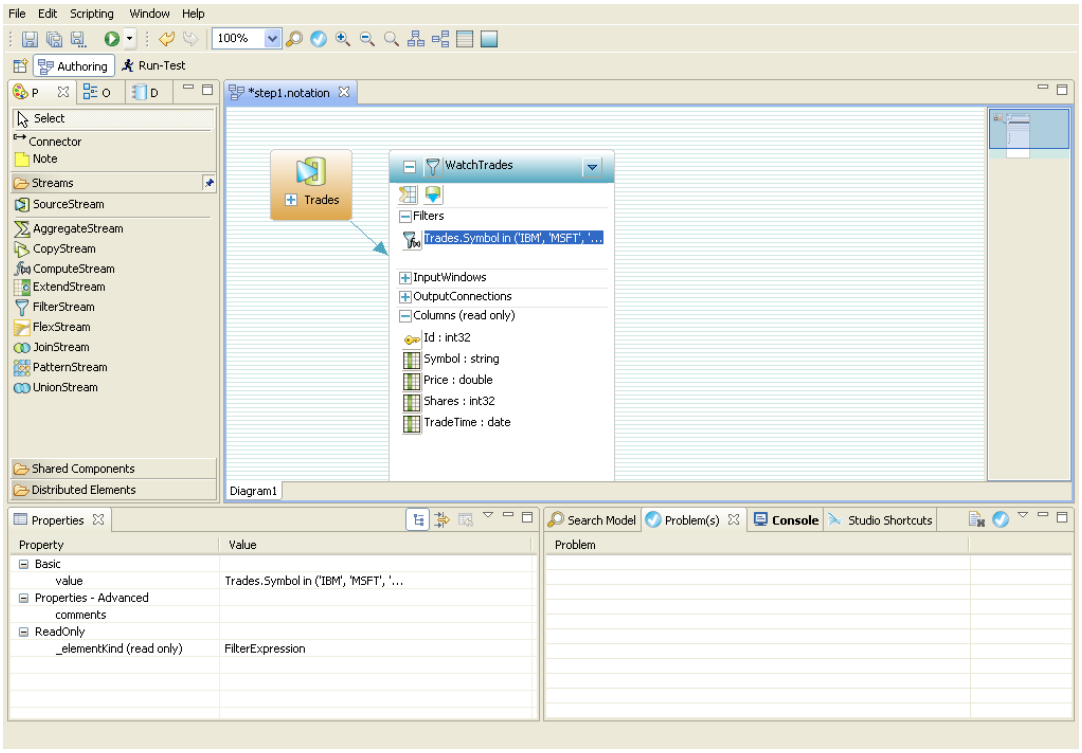
   The diagram for the new data model includes a new stream, named **WatchTrades.**

3. Click on the + on the **WatchTrades** icon in the diagram to display information about the Trades stream on the diagram.



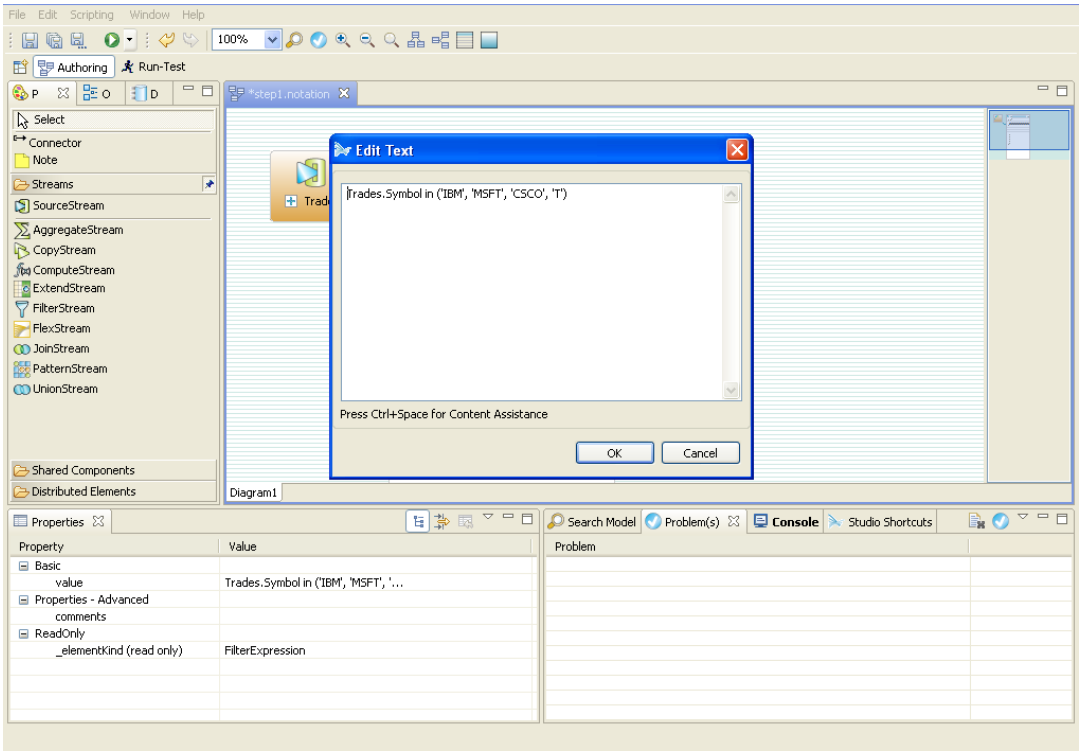
Notice that the columns listed in the **Columns (read-only)** section are the same as the Trades stream.

4. Click on the + to the left of **Filters** to open the Filters sub-compartment.



Click the **filterExpression** row to highlight it. The "**...**" button appears in the row.

Click this button to bring up an editing window that shows the entire filter expression for this Filter Stream.



Each row of data that comes into the Filter Stream is tested using the condition defined in the filter expression. Only those rows that satisfy the condition are passed on through the Filter Stream.

The condition defined for this Filter Stream:

```
Trades.Symbol in ('IBM', 'MSFT', 'CSCO', 'T')
```

In this expression, `Trades.Symbol` specifies the value of the *`Symbol`* column from a row originating in the Trades stream.

The keyword `in` introduces a list of values to which the value on the left side will be compared. The condition is satisfied only if the value from incoming row is one of those in the list.

The Filter Stream accepts one row of data at a time from the Trades stream. From each row, the Filter Stream extracts the value in the *`Symbol`* column and checks whether it matches one of the values in the list. If it does, the row is passed on through the Filter Stream; otherwise it is not retained in this stream.

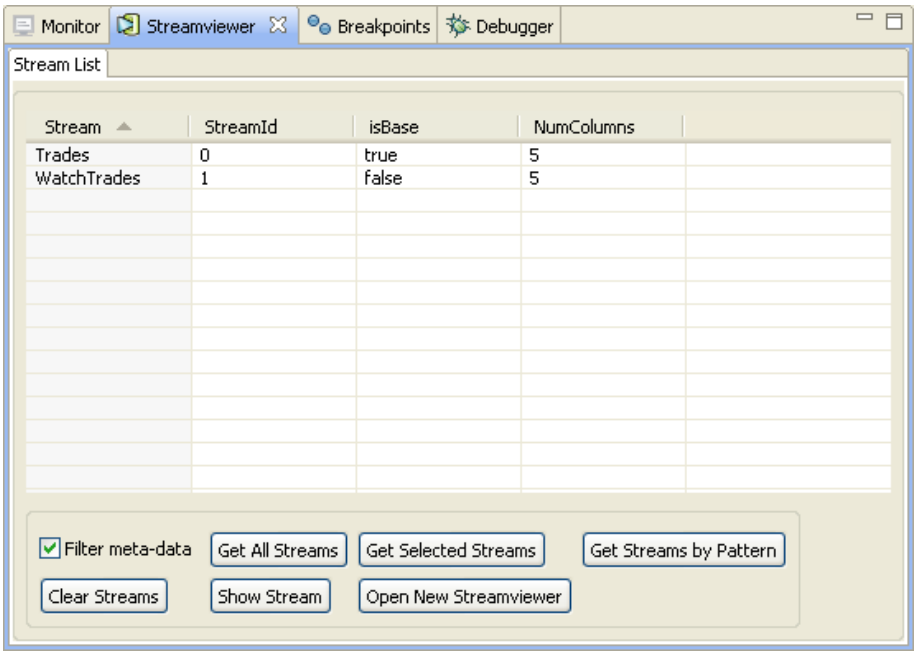
5. Close the editing window.

Test the Filter Stream

6. Click the green arrow button under the **Scripting** menu. This will start the Sybase Aleri Streaming Platform with the data model. After a moment, the green arrow button should turn into a red box.

7. Bring up the **Streamviewer** on the right side of the **Run-Test perspective**. See the section called "Set up the Streamviewer" for more detailed instructions.

   Check the **Filter meta-data** check box, then click **Get All Streams** to see this data model's list of streams. This model contains the Trades stream, as the previous model did, and also the WatchTrades stream.



8. Double-click on the stream name to display the **StreamData-Trades** tab panel.

   Click the **Viewer** tab to bring that sub-panel to the front.

   Double-click on the **WatchTrades** row to display a **StreamData-WatchTrades** tab panel.

Upload and filter the test data

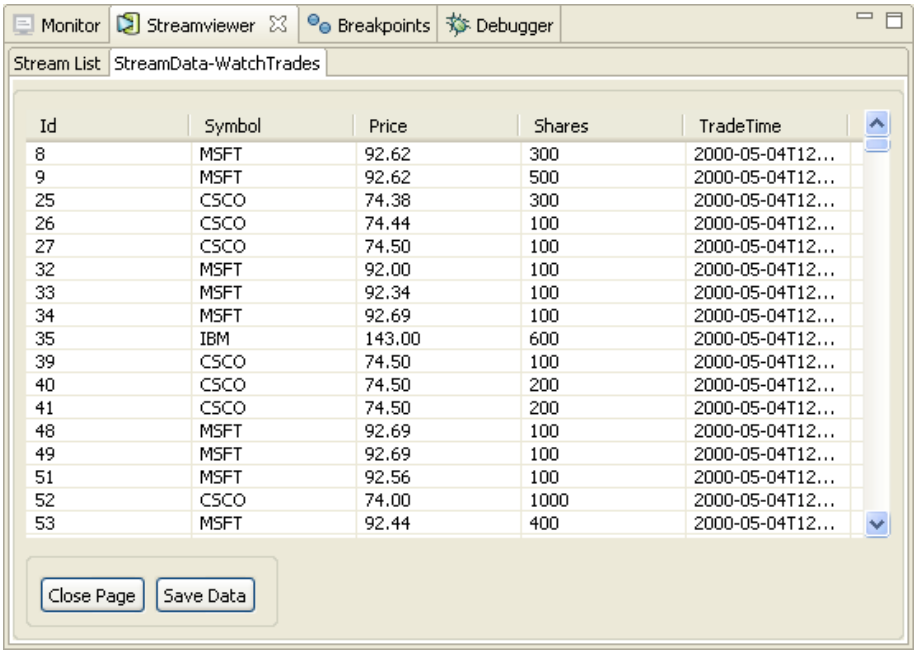
9. Click the Command Execution **Upload** tab to bring this sub-panel to the front.

   If the entry for `Trades.xml` is in the **Input File(s)** list, click this entry once to highlight it. Otherwise, click **Browse** and browse to this file, as before.

   Make sure the **Convert** check box is checked.

   Click **Upload** to upload the trade records from `Trades.xml` to the Sybase Aleri Streaming Platform. The records are received into the Trades stream, which passes all the records to the WatchTrades stream. The WatchTrades stream retains only the rows that satisfy the condition defined in the *`filterExpression`*.

10. To verify the effect of the Filter Stream, scroll through the records in the **StreamData-WatchTrades** sub-panel. The only values displayed in the **Symbol** column are `IBM`, `MSFT`, `CSCO` or `T`.



Id	Symbol	Price	Shares	TradeTime
8	MSFT	92.62	300	2000-05-04T12...
9	MSFT	92.62	500	2000-05-04T12...
25	CSCO	74.38	300	2000-05-04T12...
26	CSCO	74.44	100	2000-05-04T12...
27	CSCO	74.50	100	2000-05-04T12...
32	MSFT	92.00	100	2000-05-04T12...
33	MSFT	92.34	100	2000-05-04T12...
34	MSFT	92.69	100	2000-05-04T12...
35	IBM	143.00	600	2000-05-04T12...
39	CSCO	74.50	100	2000-05-04T12...
40	CSCO	74.50	200	2000-05-04T12...
41	CSCO	74.50	200	2000-05-04T12...
48	MSFT	92.69	100	2000-05-04T12...
49	MSFT	92.69	100	2000-05-04T12...
51	MSFT	92.56	100	2000-05-04T12...
52	CSCO	74.00	1000	2000-05-04T12...
53	MSFT	92.44	400	2000-05-04T12...

11. Click the red stop button to stop the Sybase Aleri Streaming Platform.

You're now ready for the the next step in this tutorial, which introduces a new stream in the data model — one that will aggregate the data filtered by WatchTrades.

- To author this new stream yourself, go to Chapter 5, *Authoring an Aggregate Stream*.
- To skip the process of authoring the new stream by loading a new version of the data model that has the stream already in it, go to Chapter 6, *Aggregating Streaming Data*.

Chapter 5. Authoring an Aggregate Stream

Authoring an Aggregate Stream is more complicated than authoring a Filter Stream. It requires creating a new element in the data model and adding all of its attributes.

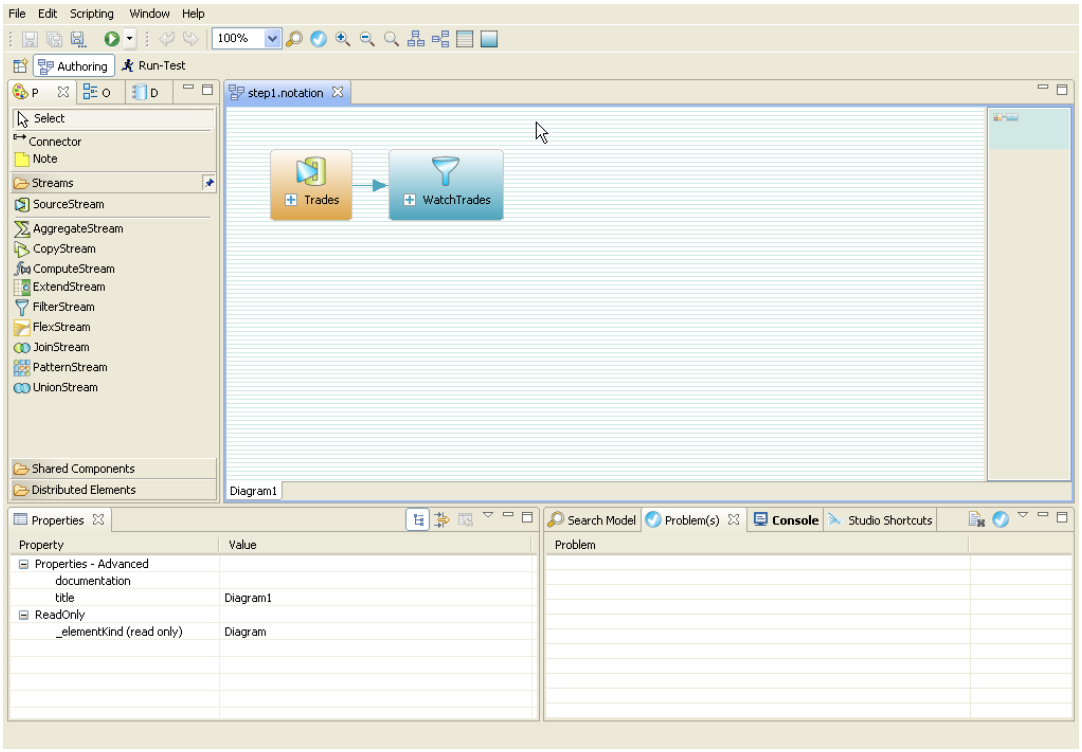
Get started

1. Make sure that:

   - The Aleri Studio is open.
   - The `step1.notation` file has been loaded.

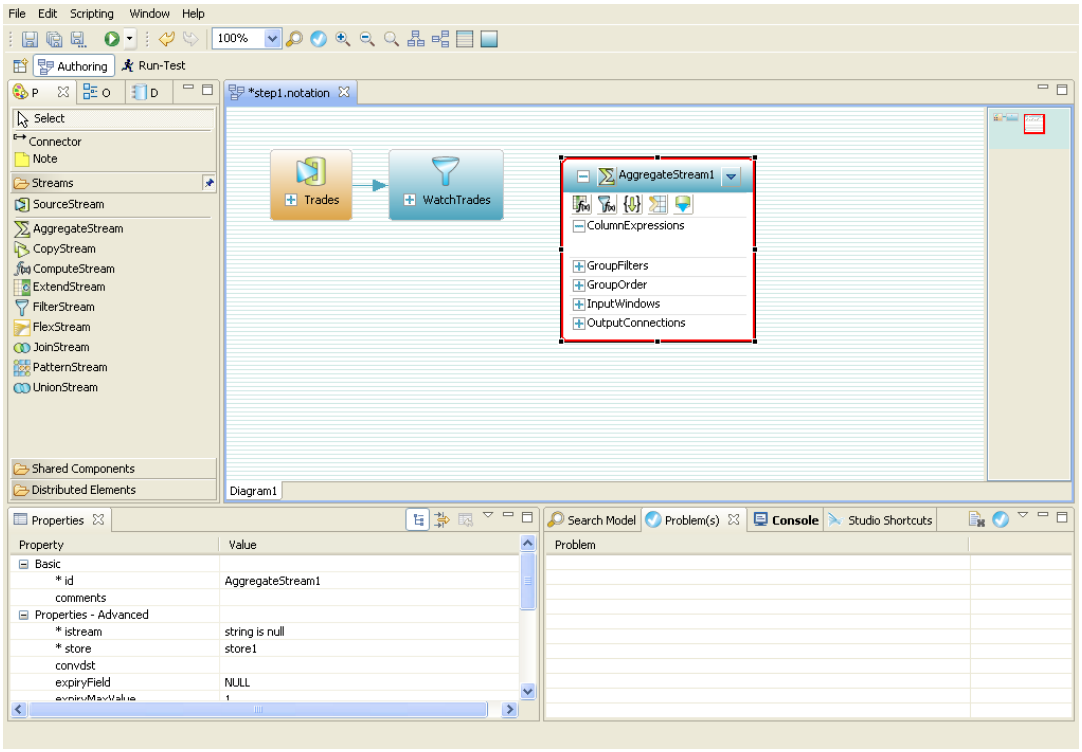
     See the section called “Load the first model” for instructions.

   - The Sybase Aleri Streaming Platform is not running.



2. Click **Aggregate Stream** in the **Palette** tab panel on the left side of the screen.

   Click once in the diagram area (at about the same height as the top of the **Trades** shape). A new Aggregate Stream appears in the diagram of the Compartment mode:



3. Click on *`AggregateStream1`* in the second compartment of the stream shape to highlight the text.

4. Type the name `WatchTradesWithAverage` in this compartment, then press **Enter** to set the new name.

Connect the Aggregate Stream to its input stream

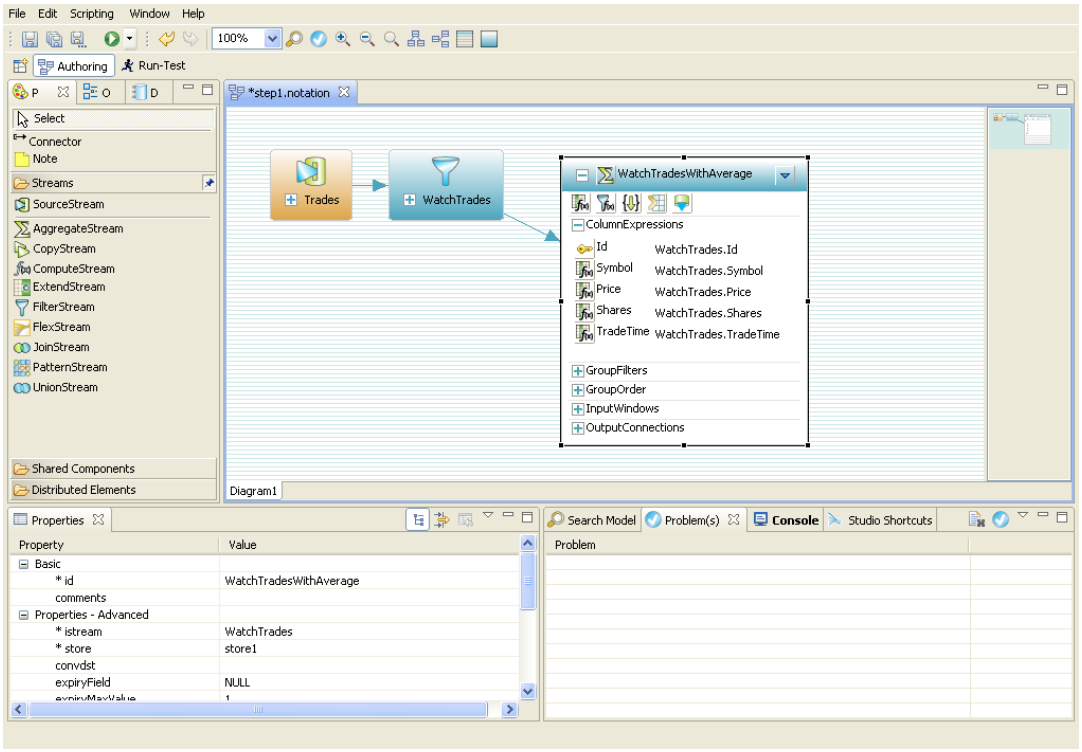
5. Click on **Connector** in the **Palette** tab panel.

   Move the mouse cursor over the diagram area. The cursor arrow now has a “plug” graphic attached to its tail.

6. Click once on the **WatchTrades** shape, then move the cursor over to the new **WatchTradesWithAverage** shape.

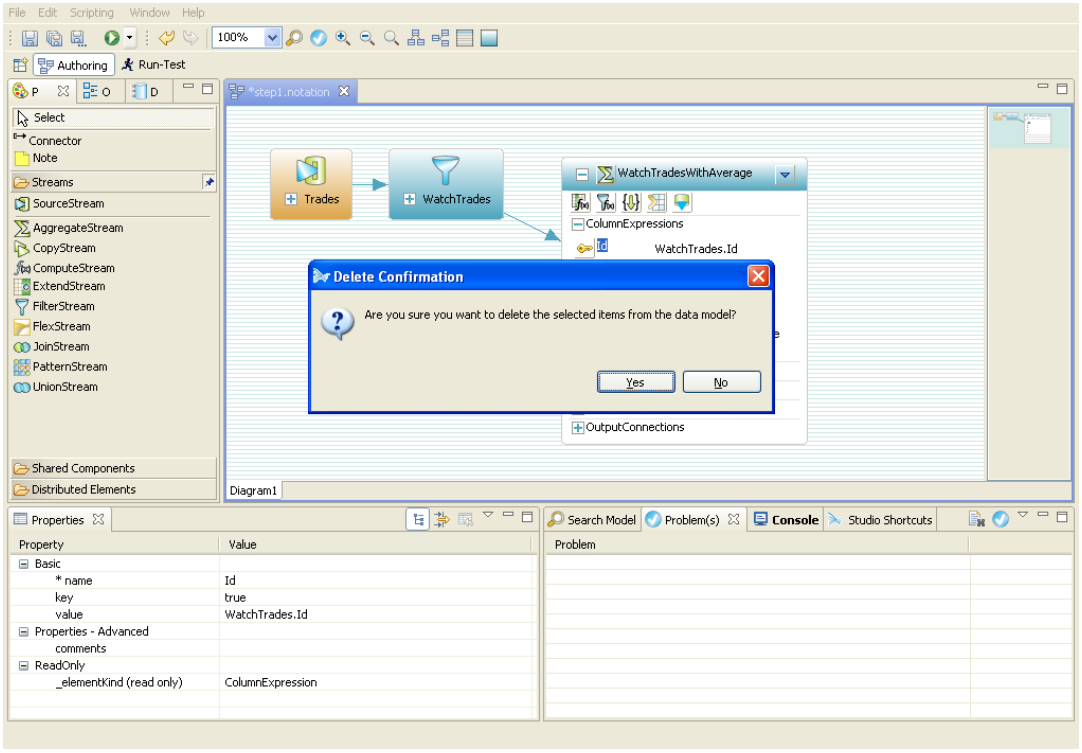
   Click once on the **WatchTradesWithAverage** shape.

   - An arrow appears from the **WatchTrades** shape to the **WatchTradesWithAverage** shape.
   - New default values appear in the **ColumnExpressions** sub-compartment.



Fix the ColumnExpressions and set the key

7. The Id column is not needed for this stream. Click on Id, then press the **Delete** key. The following dialog appears:



Click on **Yes**.

Click once on the icon to the left of Symbol. This turns the column into a key field.

Edit the Column Expressions

The last step is to edit the column expressions. These tell the Aggregate Stream how to calculate the values in each column of each of its output records.

8. Look at the entries in the **ColumnExpressions** sub-compartment of the **WatchTradesWithAverage** stream shape.

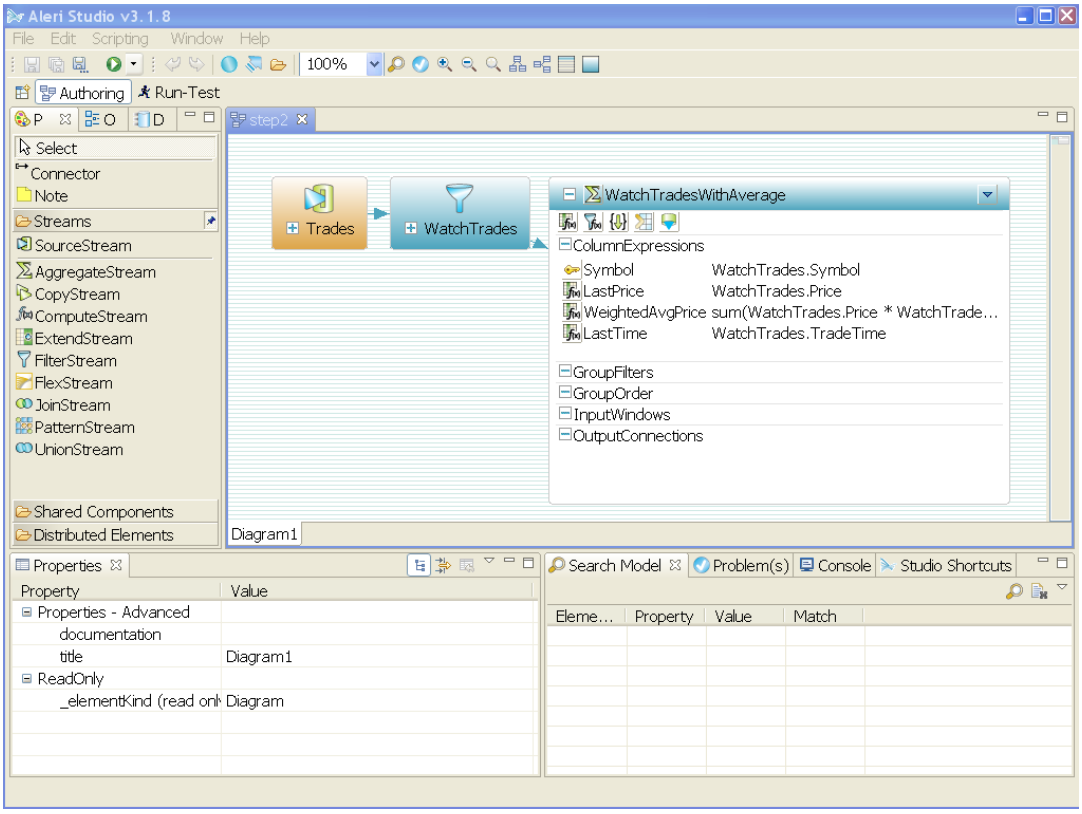
   At this point, each expression is just a reference to a column in the input stream. These expressions specify that the values in the output rows are simply brought in from the input stream with no further calculation or processing.

   You want to change the names of the columns, and also make the third column (`WeightedAvgPrice`) compute a weighted average for each group. To make this happen, you must edit the column expressions.

9. In the ColumnExpressions sub-compartment, double-click on the text Price. Change it to “LastPrice”, and press **Enter** to save the new name.

   In the ColumnExpressions sub-compartment, double-click on the text TradeTime. Change it to “LastTime”, and press **Enter** to save the new name.

   In the ColumnExpressions sub-compartment, double-click on the text Shares. Change it to “WeightedAvgPrice”, and press **Enter** to save the new name.



10. Double-click on the expression to the right of WeightedAvgPrice. The Expression Editor window appears, showing the current column expression.

11. Delete the current entry and type in the following expression:

```
sum(WatchTrades.Price * WatchTrades.Shares)/ sum(WatchTrades.Shares)
```

    Close the editing window.

    For each group of records created by the group expression, this expression does the following:

    - For each row, multiplies the Price value by the Shares value.
    - Adds all these values together to derive a total value for all shares in this group.
    - Divides this sum by the total number of shares for the group.

    The result is a volume-weighted average price for each group and Symbol.

12. Select **Check Syntax** from the top tool bar (it is a blue check mark). The Aleri Studio checks the expression just typed in. If there are no problems with the expression, the **Syntax Check Succeeded** message appears.

    Click **OK** to close the message box.

13. Close the editing window. The ColumnExpression should then show the expression you just saved.

14. Select **Save As...** from the **File menu**.

    Save the revised data model under a new name.

15. Exit the Aleri Studio by selecting **Exit** from the **File menu**.

You are now ready to stream data through the augmented data model and see the aggregated results.

Chapter 6. Aggregating Streaming Data

The stream type that was added to the data model in the last chapter is the Aggregate Stream. This part of the tutorial will show how an Aggregate Stream calculates the weighted average of streaming market data.

An Aggregate Stream's input comes from exactly one other stream. The Aggregate Stream assigns each incoming record to a group, based on values from some of the columns. The Aggregate Stream's output is one record for each group. This record can contain values that are computed across all members of the group.

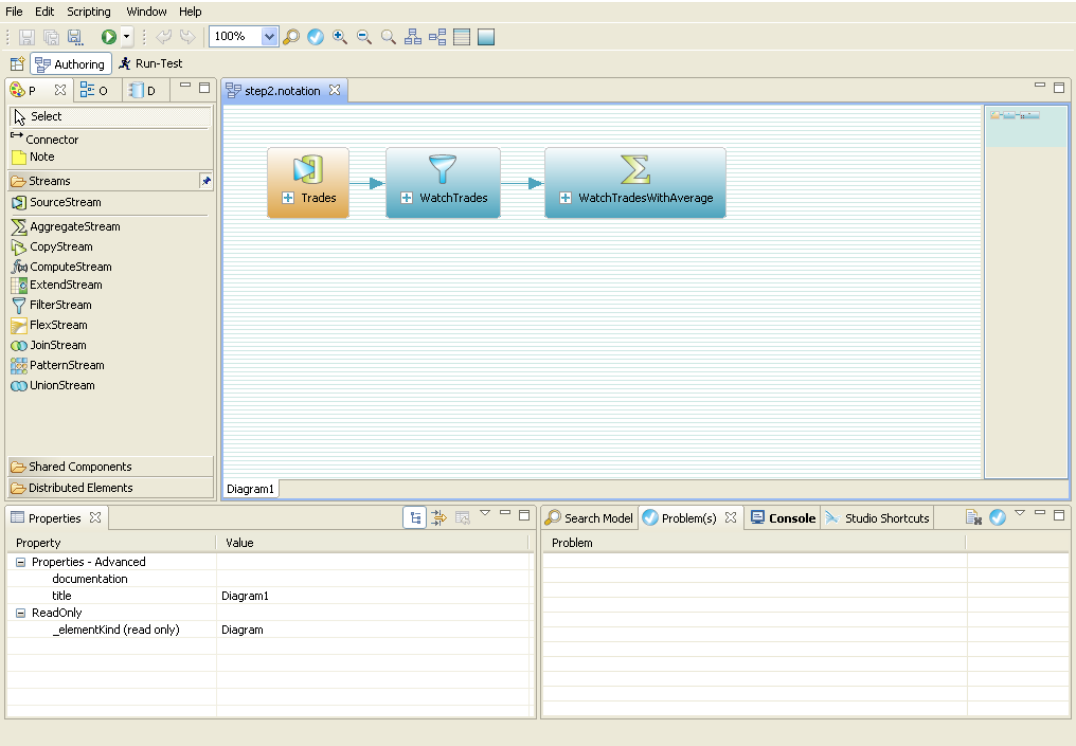
Load the third data model

Start by making sure that the Aleri Studio is running, but not the Sybase Aleri Streaming Platform.

1. Start this part of the tutorial as follows:
   - If the Aleri Studio is already open, make sure the Sybase Aleri Streaming Platform is not running. If it is, click the red stop button to stop it.
   - If the Aleri Studio is not open,
     - Start the Aleri Studio.
     - Click the **AleriStudio** button in the lower right corner of the Welcome Page, and the main Aleri Studio window appears.
2. Select **Open** from the **File** menu.

   In the **Open** dialog box, browse to the folder containing the `.notation` files for the tutorial.

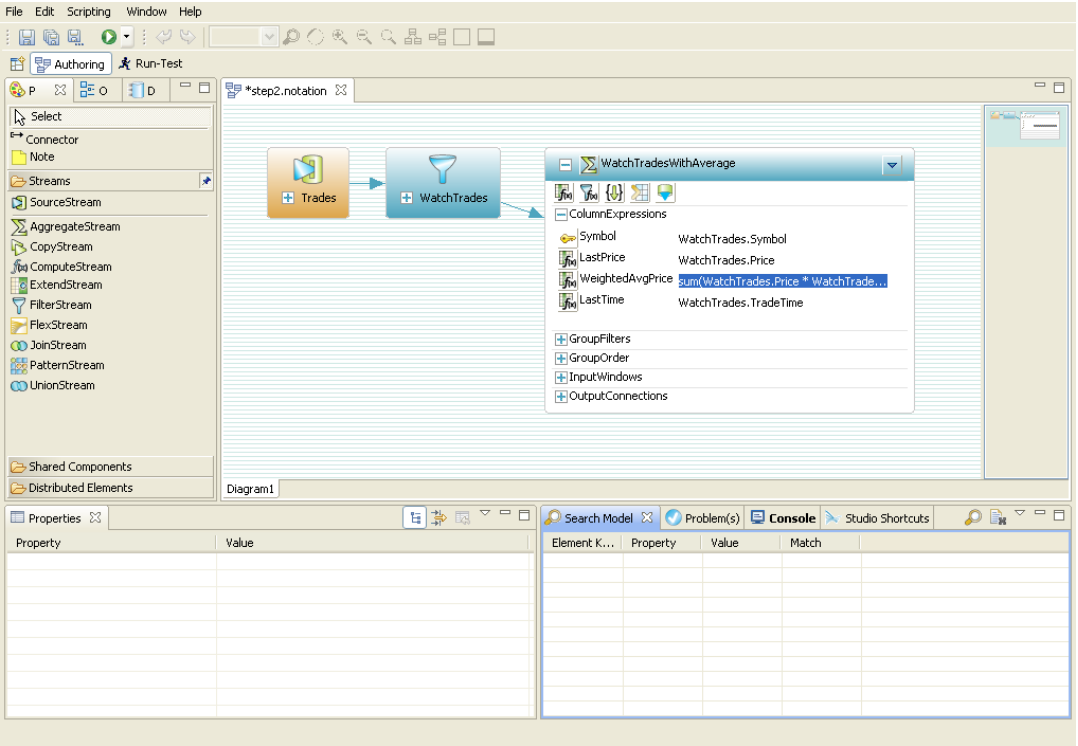
   Select `step2.notation` and click **Open** to load the `step2` data model into the Aleri Studio, and display in the Authoring perspective.



The diagram for the Aggregate data model includes a new stream, named **WatchTradesWithAverage**.

The Aggregate Stream assigns incoming records to groups and outputs one set of values for each group. To use an Aggregate Stream, you must specify expressions that compute column values for each group. (This can be a simple expression that outputs the most recently received value in that group, or a more complex calculation that applies such functions as `sum`, `max`, `min`, and `avg`).

3. Click the + on **WatchTradesWithAverage**, and, if a + is to the left of **ColumnExpressions**, click the + to open the **ColumnExpressions** sub-compartment.



There is one column expression for each of the Aggregate Stream's columns. Each expression defines how that column's value is derived from the incoming values.

Look at the first, second and third entries in the **ColumnExpressions** sub-compartment to verify the following entries:

- First row (*`Symbol`* column): `WatchTrades.Symbol`
- Second row (*`LastPrice`* column): `WatchTrades.Price`
- Last row (*`LastTime`* column): `WatchTrades.TradeTime`

These column expressions configure each of these columns in the WatchTradesWithAverage stream to receive data straight from the specified columns in the WatchTrades stream, with no other processing.

4. Double-click the third expression to the right of WeightedAvgPrice. An editing window will appear. The expression computes a value as follows:

   - For each row in the group, multiply the Price value by the Shared value.
   - Compute the sum of these derived values for all rows in the group.
   - Divide this sum by the sum of the Shares values for all rows in the group.

   The result of this calculation is stored in the *`WeightedAvgPrice`* column for each row in the WatchTradesWithAverage stream.

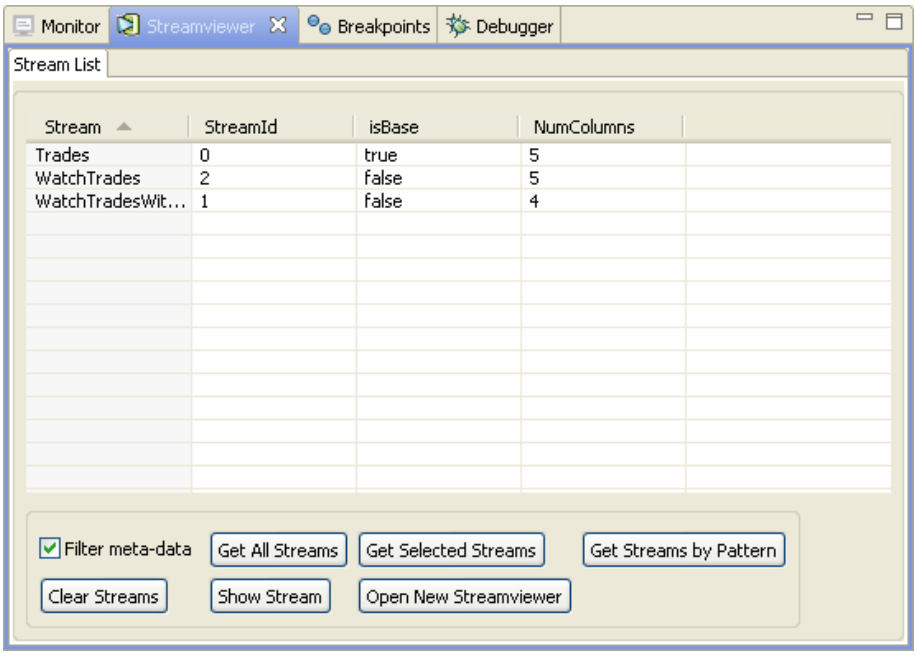
   Close the editing window.

Prepare to upload test data

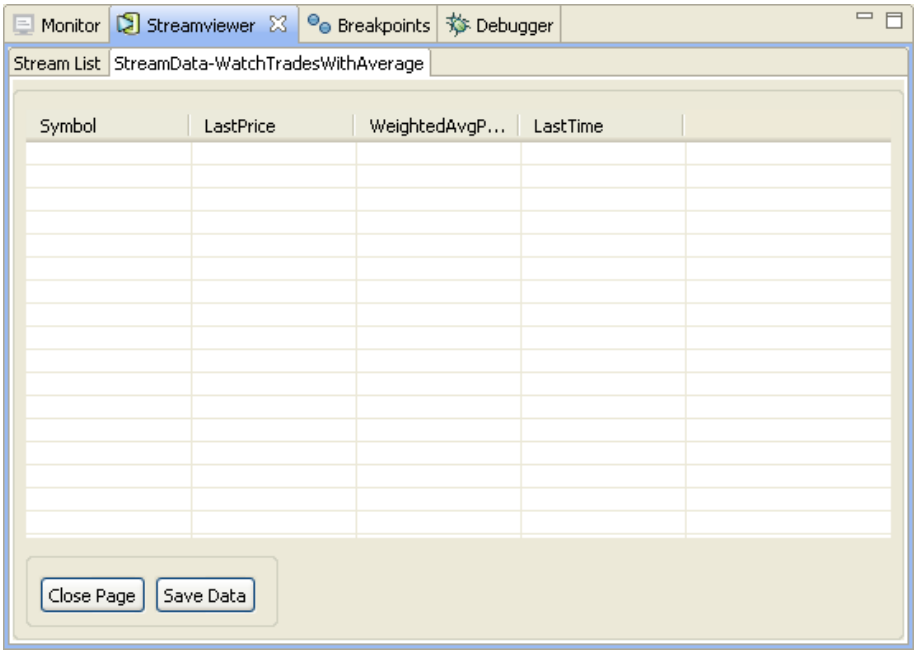
5. Click the green arrow button under the **Scripting** menu. This will start the Sybase Aleri Streaming Platform with the data model. After a moment, the green arrow button should turn into a red box.

6. Bring up the **Streamviewer** on the right side of the **Run-Test perspective**. (See the section called "Set up the Streamviewer" for detailed instructions.)

   Check the **Filter meta-data** check box, then click **Get All Streams** to see this data model's list of streams. This data model contains the Trades and WatchTrades streams, as the previous model did, and also the WatchTradesWithAverage stream.



7. Double-click on the stream name WatchTradesWithAverage. The **StreamData-WatchTradesWithAverage** tab panel appears.



Again, note that the column names in this sub-panel correspond to the columns defined for the WatchTradesWithAverage stream.

Upload and aggregate the test data

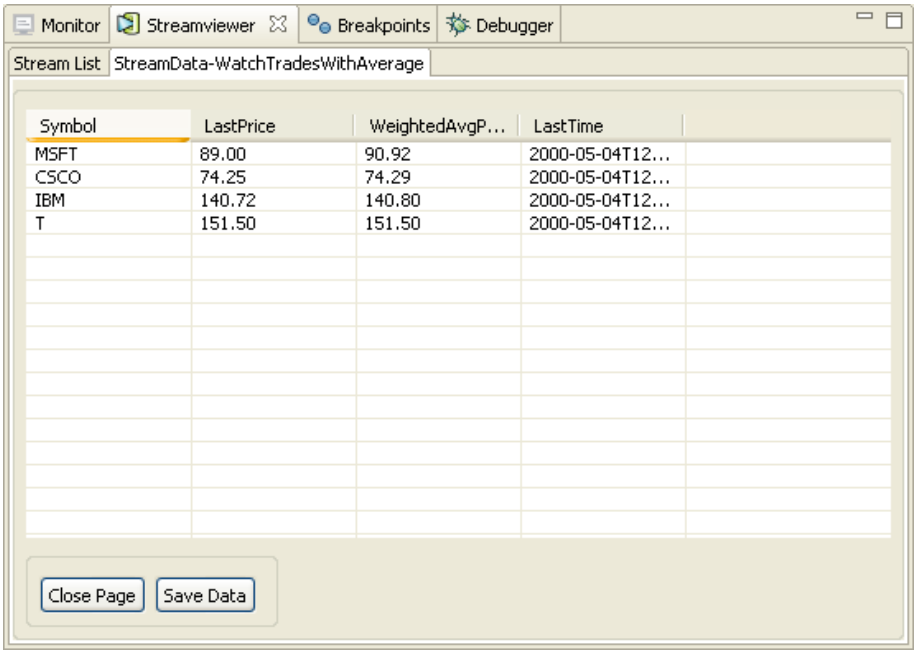
8. Click the **Upload** tab to bring this sub-panel to the front.

9. If the entry for `Trades.xml` is in the **Input File(s)** list, click this entry once to highlight it. Otherwise, click **Browse** and browse to this file, as before.

10. Make sure the **Convert** check box is checked.

11. If necessary, click the **StreamData-WatchTradesWithAverage** tab to bring this sub-panel to the front of the **Streamviewer** tab panel.

    Click **Upload** in the **Upload** sub-panel. The **StreamData-WatchTradesWithAverage** sub-panel is populated with the output from this stream.



This is what happens when the trade data is uploaded to this data model:

- The trade records from `Trades.xml` are uploaded to the Trades stream.
- The Trades stream passes all the records to the WatchTrades stream. The WatchTrades stream discards the rows that do not satisfy the condition defined in the *`filterExpression`*, and passes the rest to the WatchTradesWithAverage stream.
- The WatchTradesWithAverage stream outputs a single row for each group it creates.

Since the WatchTrades stream passes records for only four symbols, and the WatchTradesWithAverage stream groups its data by Symbol value, the output of WatchTradesWithAverage consists of only four rows.

The column expressions for the *`Symbol`*, *`LastPrice`* and *`LastTime`* columns called for values directly from corresponding output columns in the WatchTrades stream. Each time the

WatchTradesWithAverage stream receives a new row for one of its groups, the incoming *Symbol*, *LastPrice* and *LastTime* values are put into the output row for that group, replacing the values put there by the previous input row for that group. Since the *Symbol* value is always the same for all rows in the same group, the **Symbol** value in the output row will not change. However, the values in the **LastPrice** and **LastTime** columns in each row will always be the Price and Time values from the rows most recently uploaded. (When constantly processing data from a market feed, **LastPrice** always displays the current price.)

Publish individual events

Once the Sybase Aleri Streaming Platform is running, it is always ready to receive and process events. To demonstrate this, use the Aleri Studio's facility for publishing individual events to the Sybase Aleri Streaming Platform.

12. Click the **Manual Input** tab to bring this tab panel to the front.

13. From the dropdown menu in the **Source Stream** field, select **Trades**. The **Data Input** box in the **Data Input** tab panel is populated with one field for each column defined for the Trades stream.

14. Click the **Upsert** radio button in the **Event Type** box.

    ## Note:

    An `upsert` event will update an existing row if its *Id* value matches the *Id* of the input data. If there is no existing row that matches, the incoming record will be added to the stream's data as a new row.

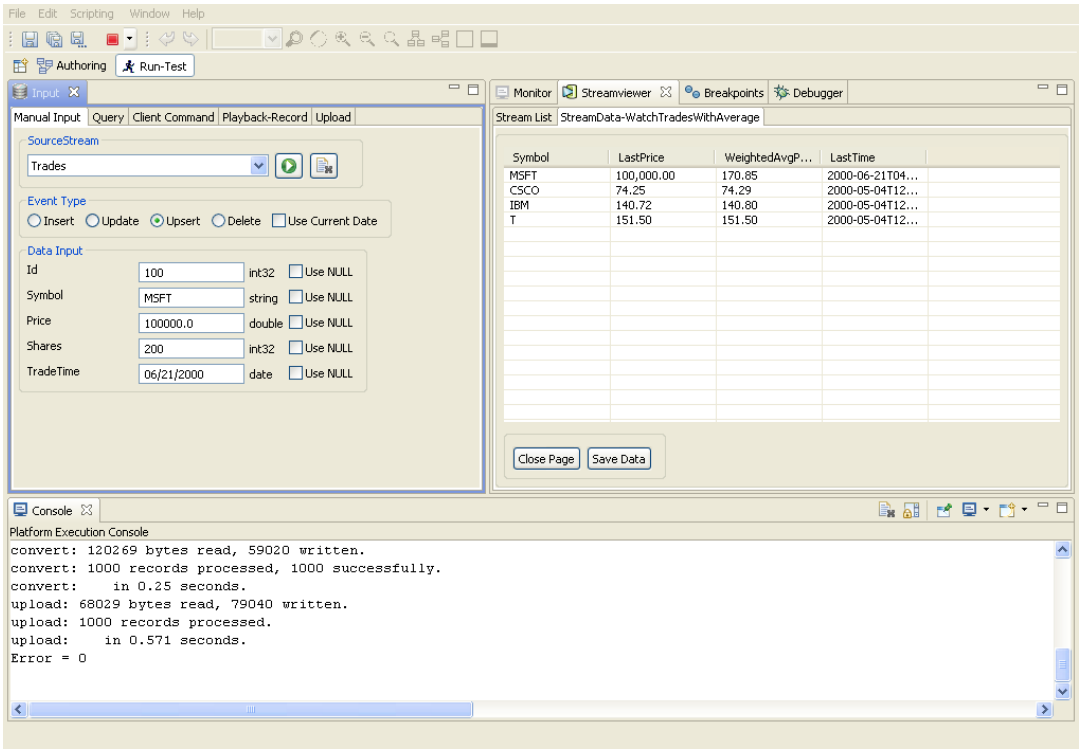
15. Make sure the value in the **Symbol** field is one of those shown in the **Symbol** column in the **StreamData-WatchTradesWithAverage** sub-panel.

    Change the values in the **Price** and **Shares** fields in the **Data Input** box. It is recommended that you type in exceptionally large values that will noticeably change the Weighted Average calculation for that symbol.

    Click the green arrow button to publish the data.

    In the **StreamData-WatchTradesWithAverage** sub-panel, the **LastPrice**, **WeightedAvgPrice** and **LastTime** values for that Symbol are updated.

    The following screen shot shows the results for some sample values:



What happens, step by step:

- The new event is published to the Trades stream.
- The Trades stream passes the new event to the WatchTrades stream. Since this event's `Symbol` value satisfies **WatchTrade**'s filter expression, it is passed on to the WatchTradesWithAverage stream.
- When it receives this new data, the WatchTradesWithAverage stream updates its `LastPrice` and `LastTime` values for this `Symbol`, and recalculates the appropriate `WeightedAvgPrice` value. The Aleri Studio instantly updates its Streamviewer display accordingly.

16. Click the red stop button to stop the Sybase Aleri Streaming Platform.

Chapter 7. Joining Data from Two Streams

The previous chapters of this tutorial demonstrated the Sybase Aleri Streaming Platform's ability to perform basic data manipulations. This chapter will take the first step toward building a data model to fulfill a business requirement.

Suppose the manager of two or more stock portfolios wants to keep track of each portfolio's total value, and how each portfolio's current value compares with a weighted average calculated over some time period. This chapter demonstrates a simple data model that can calculate these measurements and update them in real time. The Streamviewer provides a continuously updating display of these measurements.

Load the Join data model

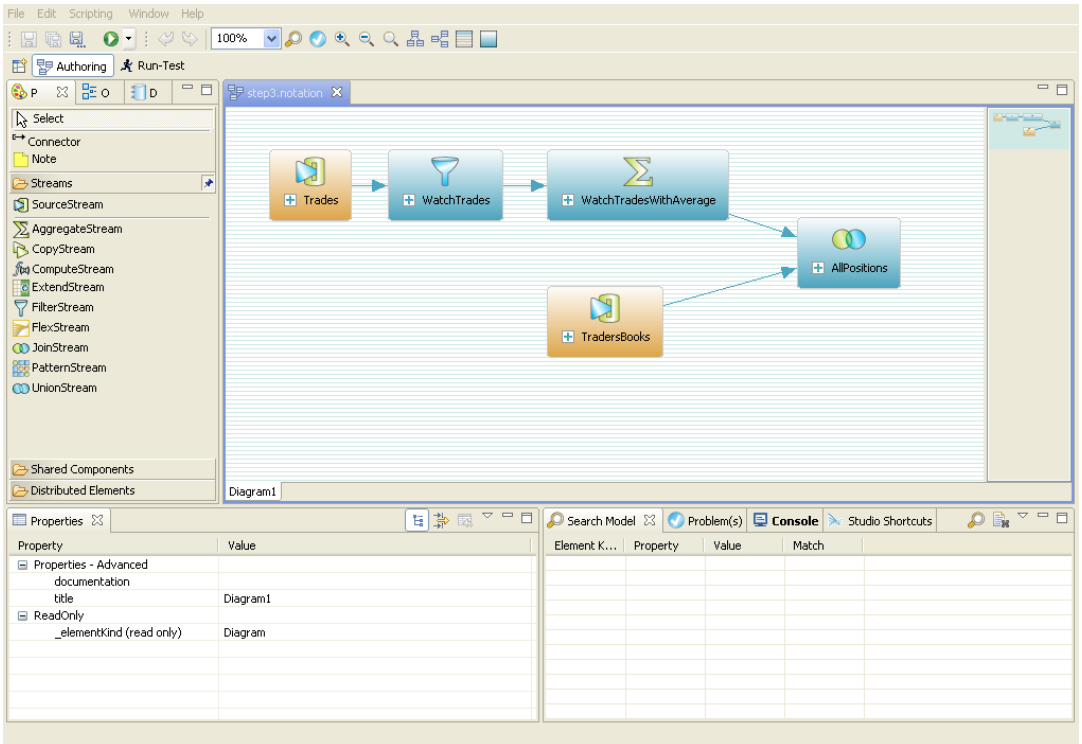
As in previous chapters, start by making sure that the Aleri Studio is running but the Sybase Aleri Streaming Platform is stopped.

1. If the Aleri Studio is already open, make sure the Sybase Aleri Streaming Platform is not running. If it is, click the red stop button to stop it.

2. Select **Open** from the **File** menu.

   Load the file `step3.notation` into the Aleri Studio.

   The `step3` data model is loaded into the Aleri Studio, and displayed in the Authoring perspective.



   This version of the data model contains the following new components:

   TradersBooks — A new source stream whose entries define each “book” (that is, which stocks are included in a particular book, and how many shares of each). TradersBooks demonstrates another use of a Source Stream: to store static or seldom changing data used in a data model's calculations.

AllPositions	A type of derived stream called a Join Stream. This type of stream is like a join of relational database tables. It combines data from two sources, using matching values in one or more common columns to decide the composition of each row. As the diagram indicates, AllPositions combines current and averaged trade prices (from Trades, through WatchTrades and WatchTradesWithAverage) with book composition information from TradersBooks.

How the Join Stream works

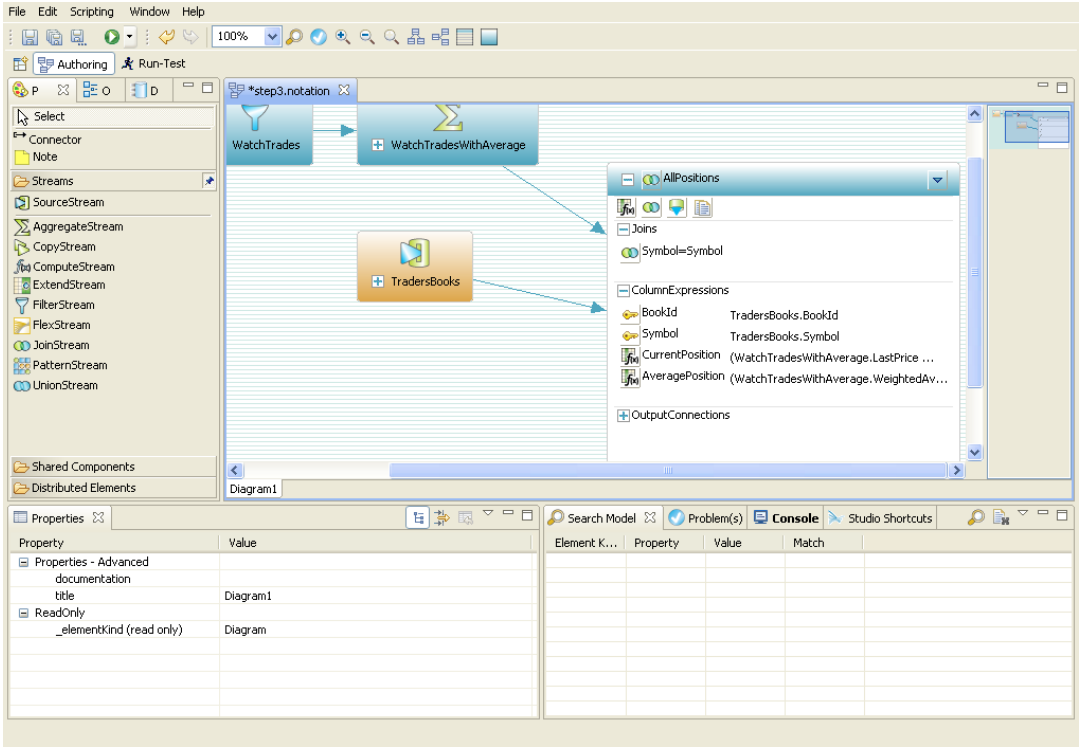
The purpose of the Join Stream is to produce a single stream from two or more input streams. Common values in each stream are used to create new records that combine information from the source records.

Our data model now has two Source Streams, Trades (the source of data on individual trades) and **TradersBooks** (entries that list the symbol and quantity of each position in each book).

3. Click on the + on the AllPositions icon, then click the + to the left of **ColumnExpressions** to open the sub-compartment.

   Each row maintained by this stream is identified by a unique combination of BookId and Symbol. The stream's column expressions calculate a CurrentPosition value and an AveragePosition value for each BookId/Symbol combination.

4. To see how these values are matched, look at the details of the “Joins”. Click the + to the left of **Joins**, and click on the row in the new sub-compartment.



   The join between these two streams is a “left join.” The two source streams are related by each one's *Symbol* column.

5. Now look at the ColumnExpressions. The expression for the CurrentPosition column is

```
WatchTradesWithAverage.LastPrice * TradersBooks.SharesHeld
```

For this column, the Sybase Aleri Streaming Platform calculates a CurrentPosition value for a Book position by multiplying the Symbol's current price (from WatchTradesWithAverage) by the number of shares of this Symbol held in a particular book (from TradersBooks).

The expression for the fourth column, *`AveragePosition`*, is

```
WatchTradesWithAverage.WeightedAvgPrice * TradersBooks.SharesHeld
```

For this column, the Sybase Aleri Streaming Platform calculates an “Average Position” value for a Book position by multiplying the Symbol's average price (as calculated by WatchTradesWithAverage) by the number of shares from TradersBooks.

To summarize: here is what happens when this model is run on the Sybase Aleri Streaming Platform:

- As market data comes in to the Sybase Aleri Streaming Platform, it is received by the Trades stream and passed to the WatchTrades stream.
- The filter expression in the WatchTrades stream passes through only those trade records whose Symbols are on its list. These are passed on to the Aggregate Stream WatchTradesWithAverage.
- The WatchTradesWithAverage stream groups the incoming Trade records by Symbol. For each group of trades, WatchTradesWithAverage keeps the most recent Price and Trade time, and computes a weighted average of all Prices.
- The TradersBooks stream receives data on the positions held in each Book. This information is passed to the Join Stream AllPositions.
- For each position defined in the TradersBooks stream, the AllPositions stream computes a *`CurrentPosition`* and *`AveragePosition`* using *`SharesHeld`* data from the TradersBooks stream and *`LastPrice`* and *`WeightedAvgPrice`* information from the WatchTradesWithAverage stream.

In the next section, you will subscribe to the AllPositions and TradersBooks streams using the Streamviewer.

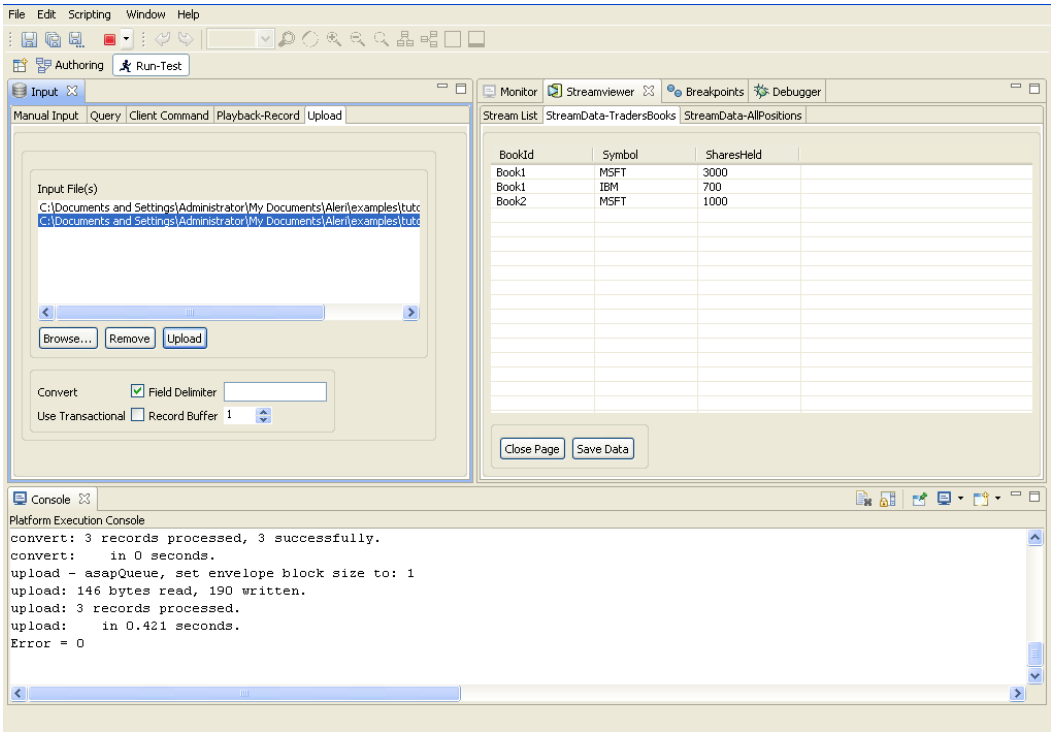
Upload and view AllPositions data

6. Start the Sybase Aleri Streaming Platform by clicking on the green start button.
7. Set up the Streamviewer as follows:
   - Bring up the **Streamviewer** and click **Get All Streams**.
   - Create **StreamData** tabs for the AllPositions and TradersBooks streams.
   - Bring the **StreamData-TradersBooks** sub-panel to the front.
   - Click the **Upload** tab on the Aleri Studio window to bring this sub-panel to the front.

- In the **Upload** sub-panel, click **Browse...** and browse to the file `TradersBooks.xml`.

  In the **Upload** sub-panel, check the **Convert** check box (if it is not already checked), then click **Upload**.

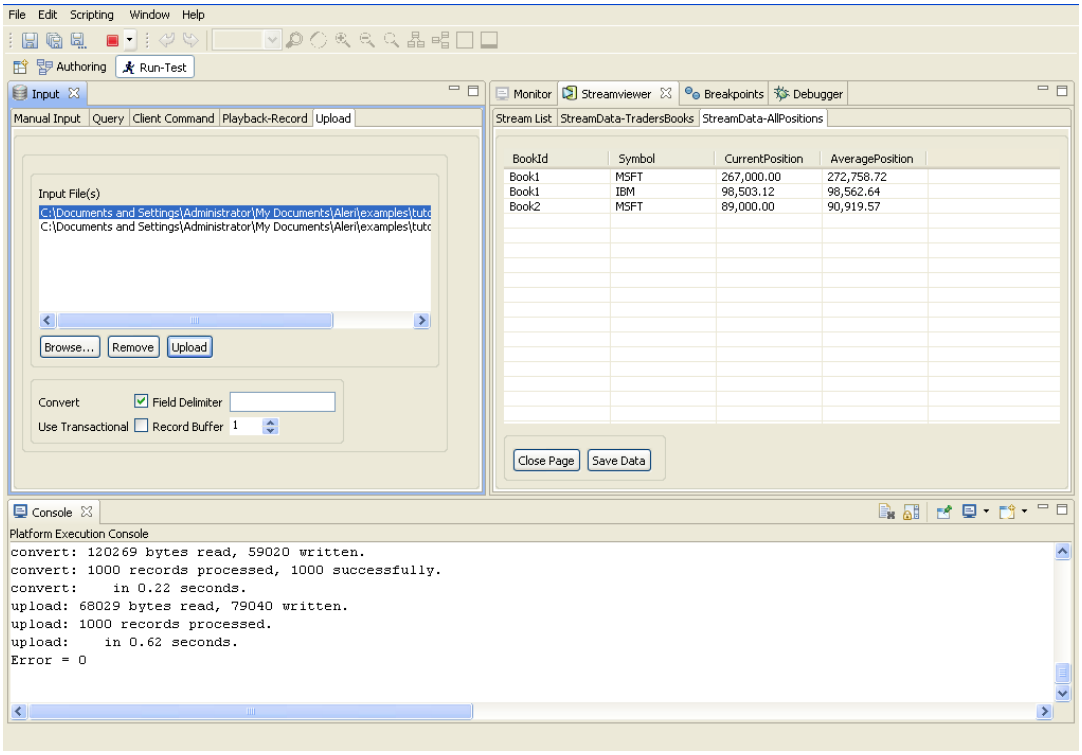
  The fields in the **StreamData-TradersBooks** contain information about the positions in each Book.



Note:

See the section called “Upload the Trade data” for more detailed instructions on uploading data from a file.

8. Click the **StreamData-AllPositions** tab to bring this sub-panel to the front.



The AllPositions stream was initialized when the Sybase Aleri Streaming Platform started up, with one row for each Book defined in the TradersBooks stream. All the values are nulls because no Trade data had been received yet.

9. Return to the **Upload** sub-panel.

   Select `Trades.xml` from the **Input File(s)** list. (If this filename is not in the list, click **Browse...** and browse to it first.)

   You can see the fields in the **StreamData-AllPositions** sub-panel fill in as the Sybase Aleri Streaming Platform processes the trade data from the uploaded file.

10. Click the red stop square under the **Scripting** menu to stop the Sybase Aleri Streaming Platform.

    The Sybase Aleri Streaming Platform is stopped. The rows of data in the **StreamData-Trades** sub-panel disappear (since the stream no longer exists on the Sybase Aleri Streaming Platform).

    Messages sent out from the Sybase Aleri Streaming Platform as it shuts down appear in the **Aleri Server Console** tab panel.

Congratulations! You have successfully completed the Aleri Tutorial in which you learned how to use the Aleri Studio to compose a model and run it with sample data.

The Aleri Studio provides other example models as well. These models are found in the File/Open menu of the Aleri Studio, which appears when you click on **Open Example** on the Welcome page. On a Windows PC, you would then select the `My Documents\Aleri` folder. On Linux and Solaris systems, it is `/home/aleri/aleri/platform/3.2.0/examples directory` unless you specified an alternate location when you installed the software.