Oracle® Fusion Middleware

Using Visualizer for Oracle Event Processing

12*c* Release (12.1.3)

E28541-03

October 2015

How to use Oracle Event Processing Visualizer to build, configure, deploy, and manage Oracle Event Processing applications that run in single-server and multiserver domains. This document also provides information about how to edit Oracle CQL queries at run time.

Oracle Fusion Middleware Using Visualizer for Oracle Event Processing, 12c Release (12.1.3)

E28541-03

Primary Authors: Madhubala Ponnekanti, Oracle® Corporation

Contents

10 Jetty Configuration

11 JMX Configuration

12 Data Source Configuration

13 HTTP Publish-Subscribe Servers

14 Event Inspector Service

15 Event Type Repository

16 Persistent Event Store

17 Multiserver Domains

Preface

This document describes how to use the Oracle Event Processing Visualizer to manage Oracle Event Processing applications and servers at runtime.

Audience

This document is intended for all users of Oracle Event Processing.

Related Documents

For more information, see the following:

- *Developing Applications for Oracle Event Processing*
- *Getting Started with Oracle Event Processing*
- *Getting Started with Oracle Edge Analytics*
- *Schema Reference for Oracle Event Processing*
- *Customizing Oracle Event Processing*
- *Administering Oracle Event Processing*
- *Customizing Oracle Event Processing*
- *Developing Applications with Oracle CQL Data Cartridges*
- *Oracle CQL Language Reference for Oracle Event Processing*
- *Java API Reference for Oracle Event Processing*
- *Java API Reference for Oracle Edge Analytics*
- *Using Oracle Stream Explorer*
- *Getting Started with Oracle Stream Explorer*
- *Oracle Database SQL Language Reference* at: `http://docs.oracle.com/cd/E16655_01/server.121/e17209/toc.htm`
- SQL99 Specifications (ISO/IEC 9075-1:1999, ISO/IEC 9075-2:1999, ISO/IEC 9075-3:1999, and ISO/IEC 9075-4:1999)
- Oracle Event Processing Forum: `http://forums.oracle.com/forums/forum.jspa?forumID=820`.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in This Guide

This guide has been updated for the 12c Release and the product renamed to Oracle Event Processing. File names, screen shots, and text is updated to reflect new file names and file system structure.

1

Oracle Event Processing Visualizer

Oracle Event Processing Visualizer is a browser-based tool that enables you to view, develop, configure, and monitor aspects of Oracle Event Processing applications. You can also administer the Oracle Event Processing servers and domains.

This chapter includes the following sections:

1.1 Use Visualizer to Perform Tasks

You can use Visualizer to perform the following tasks:

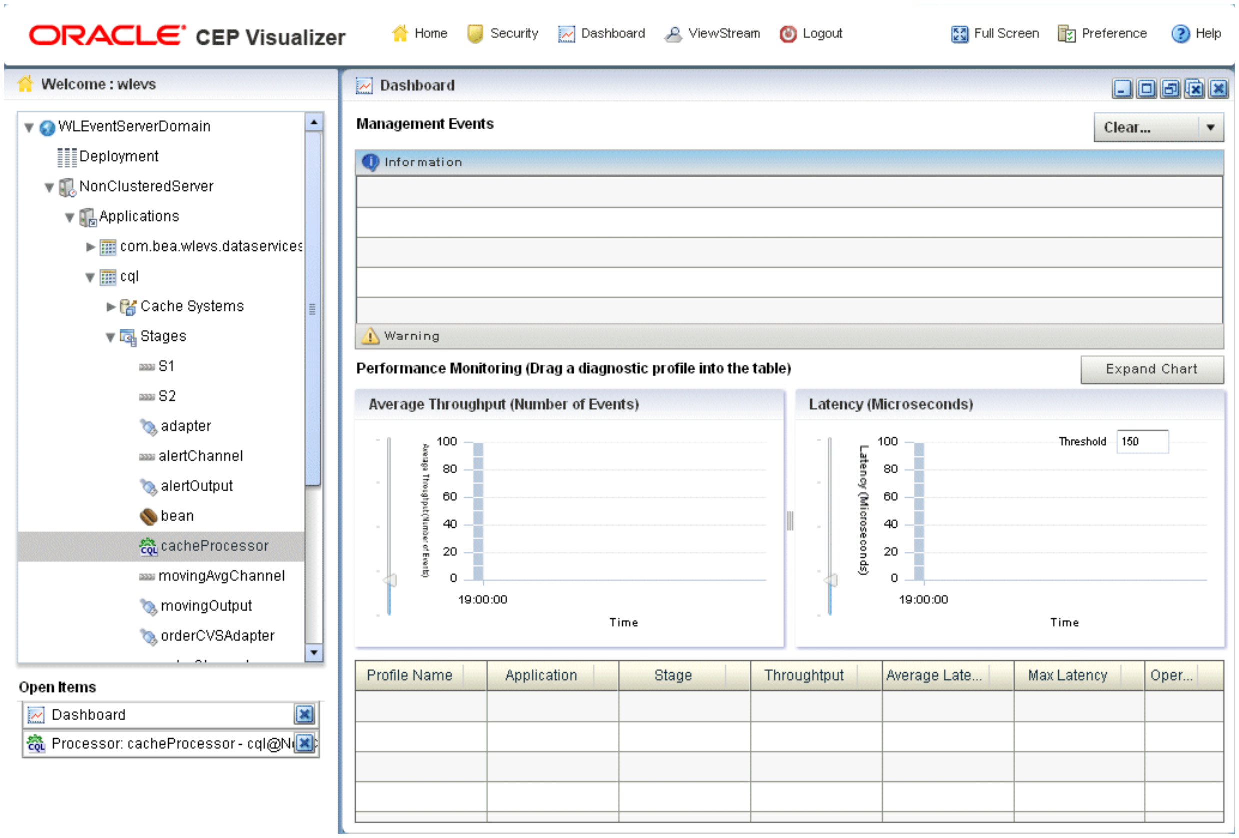
- View the structure of an Oracle Event Processing domain
- Manage security
- Configure Oracle Event Processing server instances
- Install, uninstall, suspend, resume, and update applications
- View, update, create, and delete Oracle CQL rules
- View the EPN associated with an application
- Tune application parameters and monitor application status
- Record and playback of events flowing through the EPN.
- Create diagnostic profile to monitor application stage latency and throughput.
- Monitor and perform diagnosis on your CQL Processor with Query Plan.
- Manage and create server-wide resources such as HTTP publish-subscribe channels and data sources.
- Dynamically turn server and application logging on and off.
- Manage Oracle Coherence cluster and server instances.

1.2 Visualizer User Interface

As Figure 1-1 shows, the Oracle Event Processing Visualizer has the following main panes:

- Top Pane
- Left Pane
- Right Pane

Figure 1-1 Sample Oracle Event Processing Visualizer Window



Profile Name	Application	Stage	Throughtput	Average Late...	Max Latency	Oper...

Figure 1-1 shows a domain that contains a single server instance called `NonClusteredServer`. The server contains four deployed applications: `com.bea.wlevs.dataservices`, `cql`, `fx`, and `signalgeneration`; the `cql` application is currently opened. The right pane contains the dashboard. The application called `com.bea.wlevs.dataservices` is associated with Oracle Event Processing Visualizer itself and is always deployed in an Oracle Event Processing server instance. Manage Applications for details.

1.2.1 Top Pane

This pane includes the most used buttons:

- **Home** button that takes you to the main Oracle Event Processing Visualizer page.

- **Security** button that takes you to the security page in which you can add or configure users and groups and map users to application roles and task roles; see Security Tasks for details.
- **Dashboard** button takes you to the performance management screen that you use to monitor the throughput and latency of a running application and its stages; see Visualizer Dashboard for more information.
- **ViewStream** button takes you to a screen from which you can monitor the messages streaming through the configured HTTP publish-subscribe channels; see Stream Visualizer (ViewStream) Panel for more information.
- **Full Screen** button fills your entire computer screen with the Oracle Event Processing Visualizer tool; press the **Esc** key to return to a normal screen.
- **Preferences** button takes you to a page where you can set user preferences, such as the language and maximum number of open panes, and accessibility settings such as restricting the maximum number of open panes to 1 and disabling Full Screen mode.
- **Help** button takes you to the task-oriented online-help hosted by the Oracle Event Processing server.

1.2.2 Left Pane

This pane displays a domain tree for the domain that includes all the objects contained in the domain, such as the Oracle Event Processing server instances, the deployed applications and services within each server instance, and domain-level security configuration.

The domain name is determined by the Oracle Event Processing server `config.xml` file domain element. For example, the domain tree is named `mydomain` if your `config.xml` file is like this:

```
<domain>
    <name>mydomain</name>
</domain>
```

The **Open Items** box in the lower half of the left pane lists the items that are currently open, making it easy to return to or close the windows after you have navigated away from them. Any open panels that require refresh are marked with a red X in the Open Items list. For more information, see Panels that Require Refresh .

1.2.3 Right Pane

This pane is a multi-document container. When you open multiple documents, the documents are overlaid one on top of the other.

This pane displays information about objects that you have clicked on in the left pane. The format of the information depends on the object; for example, if you click on a deployed application in the domain tree in the left pane, the right pane shows general information about the application (General tab) as well as various visual representations of the event processing network of the application (Event Processing Network tab). If you click on a particular stage of the network, such as a stream or processor, the right pane shows general information about it as well as stage-specific information, such as the rules for a processor. For more information, see Event Processing Network.

Any open panels that require refresh are marked with a red X in the title of the panel. For more information, see Panels that Require Refresh .

Table 1-1 describes the buttons in the top right corner of the right pane that you can use to manage panels. Note that these buttons are not visible if you set the **Max Open Panels** preference to 1 (see Manage User Preferences).

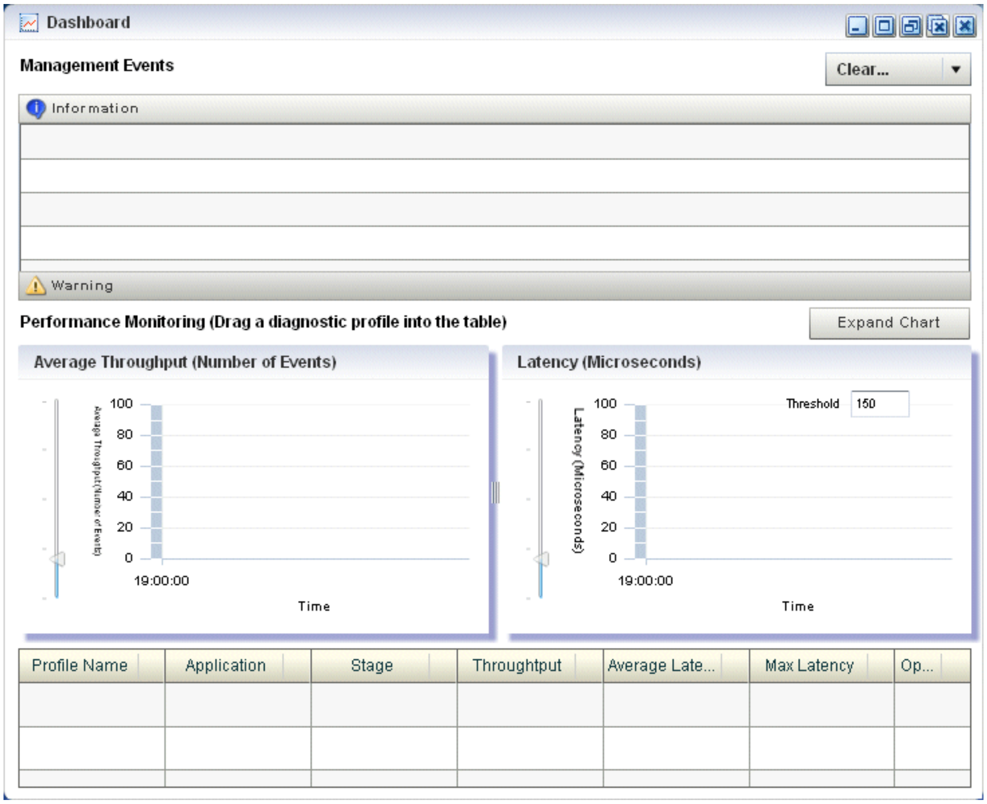
Table 1-1 Panel Controls

Button	Description
🗕	Minimize panel.
🗖	Maximize panel to full window size.
🗗	Maximize panel.
☒	Close all panels.
☒	Close panel.

1.2.4 Visualizer Dashboard

Figure 1-2 shows the Oracle Event Processing Visualizer dashboard: a performance management screen that you use to monitor the throughput and latency of a running application and its stages or a path between two stages. You get to the dashboard by clicking the **Dashboard** button in the top pane.

Figure 1-2 Oracle Event Processing Visualizer Dashboard



Profile Name	Application	Stage	Throughtput	Average Late...	Max Latency	Op...

The dashboard has two main sections:

- **Management Events**: the section at the top of the dashboard displays information and warning messages about the incoming monitoring events. Click the **Warning** bar to view the list of warning messages; click the **Information** bar to view the list of information messages.

- **Performance Monitoring**: the latency and throughput graphs display the amount of time it takes an event to pass through the specified stage or path in the EPN or the number of events passing through, respectively. The stage or path is defined in the diagnostic profile. The table at the bottom lists the available diagnostic profiles; when you click on a particular profile, the corresponding latency and throughput information is displayed in the graphs. See Monitor the Throughput and Latency of an EPN Stage or Path for details.

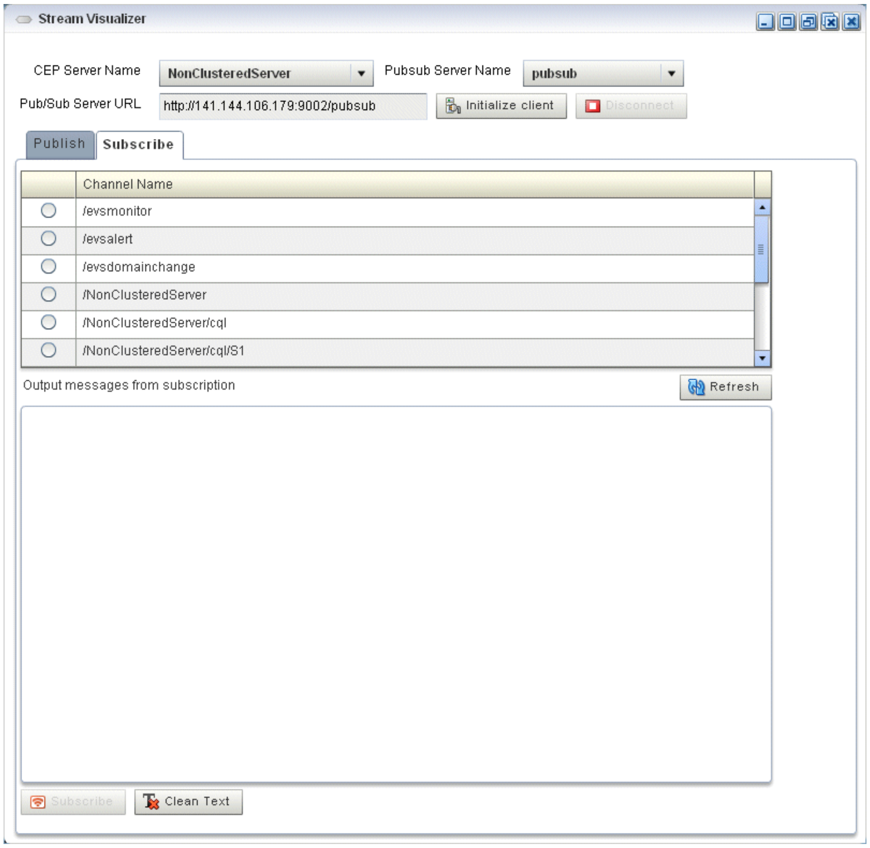
For detailed instructions on how to use this monitoring feature, see Monitor the Throughput and Latency of an EPN Stage or Path.

1.2.5 Stream Visualizer (ViewStream) Panel

The main purpose of the ViewStream panel is to allow users to watch events being published to a given HTTP publish-subscribe channel without any additional work. This is useful for debugging your application or just monitoring events flowing through the EPN.

Click the **ViewStream** button in the top panel of Oracle Event Processing Visualizer, to display the Stream Visualizer panel as Figure 1-3 shows.

Figure 1-3 Oracle Event Processing Visualizer Stream Visualizer (ViewStream)



CEP Server Name: NonClusteredServer

Pubsub Server Name: pubsub

Pub/Sub Server URL: http://141.144.106.179:9002/pubsub

	Channel Name
○	/evsmonitor
○	/evsalert
○	/evsdomainchange
○	/NonClusteredServer
○	/NonClusteredServer/cql
○	/NonClusteredServer/cql/S1

Output messages from subscription

The Pub/Sub Server URL text box displays the HTTP pub-sub server URL included with Oracle Event Processing. Click the **Initialize Client** button to start the process. You can either subscribe or publish a message to a channel using the options mentioned in this panel.

You can subscribe to a user or internal channel and view the events on it or you can publish to a user or internal channel.

For more information, see:

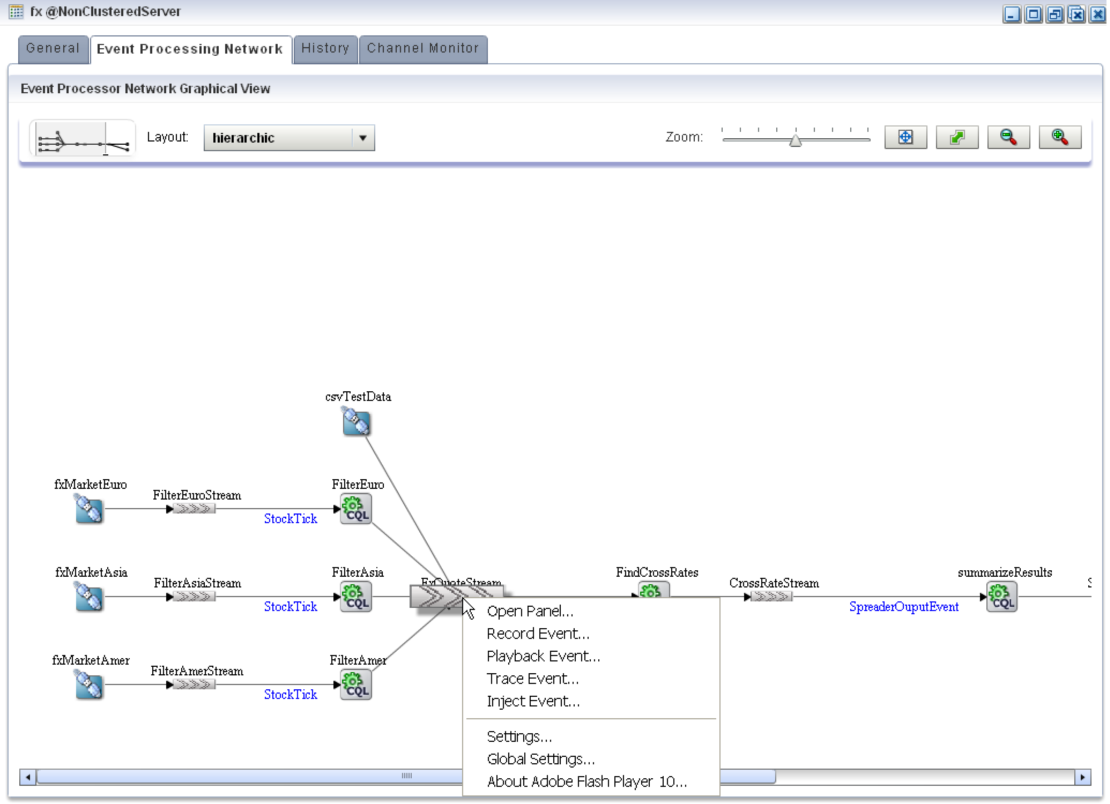
- Trace and Inject Events in the EPN
- HTTP Publish-Subscribe Servers.

1.2.6 Event Processing Network

The main purpose of the Event Processing Network (EPN) panel is to give users an overall view of the stages in an Oracle Event Processing application and the various event types they produce and consume.

In the left pane, navigate to and expand the **Applications** node, select an application, and in the right pane, click the **Event Processing Network** tab to view the EPN as Figure 1-4 shows.

Figure 1-4 Event Processing Network Panel



1.2.7 Event Processing Network Context Menu

You can right-click any stage in the Event Processing Network and select one of the options from the Event Processing Network context menu that Table 1-2 describes.

Table 1-2 Event Processing Network Context Menu

Option	Description
`Open Panel`	Opens the General tab for the selected stage. See View and Edit the Configuration of a Stage.
`Record Event`	Opens the Record tab for the selected stage. See How to Record Events.
`Playback Event`	Opens the Playback tab for the selected stage. See Play Back Events.
`Trace Event`	Opens the Trace Event tab for the selected stage. See Trace Events on a Dynamic Channel.
`Inject Event`	Opens the Inject Event tab for the selected stage. See: • Inject a Simple Event on an Event Inspector Service Dynamic Channel • Inject a JSON String Event on an Event Inspector Service Dynamic Channel

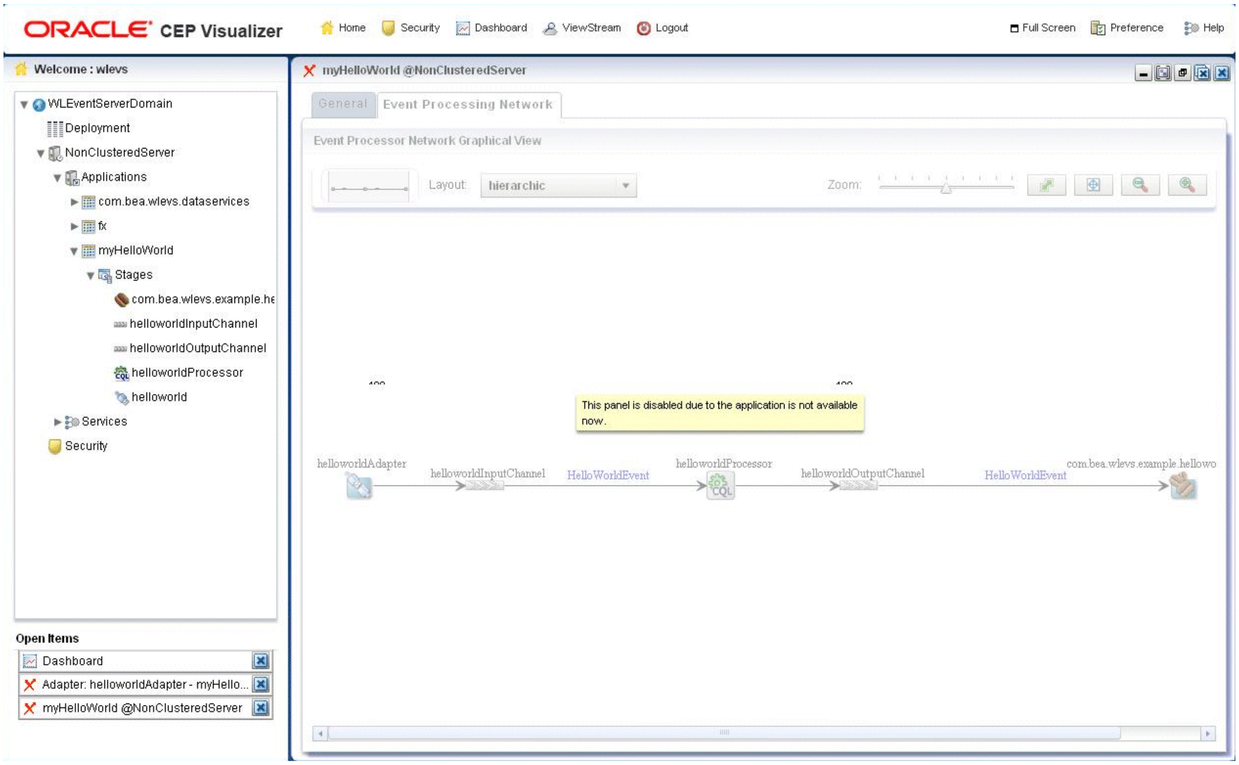
1.2.8 Panels that Require Refresh

Some operations, such as uninstalling and redeploying an application, will require you to refresh open panels.

When Oracle Event Processing Visualizer receives a notification for application re-deployment, it goes though all the affected panels that are still opened and marks the title of the panel with a red X to indicate that this panel is obsolete. To refresh such a panel, close and then re-open the panel.

For example, Figure 1-5 shows the Event Processing Network tab for an application that has been redeployed.

Figure 1-5 Event Processing Network Tab - Requiring Refresh



This panel is disabled due to the application is not available now.

1.3 Administration Tasks

This section describes the principle administration tasks you can perform using the Oracle Event Processing Visualizer, including:

- Manage Applications
- Manage Servers
- Manage Security
- Update Configuration Data

1.3.1 Manage Applications

Using Oracle Event Processing Visualizer, you can monitor and configure applications you develop and deploy as well as applications that Oracle Event Processing server deploys such as the `com.bea.wlevs.dataservices` application. You can monitor and configure the Event Processing Network (EPN), application lifecycle, and Oracle CQL rules.

For more information, see:

- Application Tasks
- Manage Applications

The `com.bea.wlevs.dataservices` application, called dataservices for short, is internal to Oracle Event Processing Visualizer and is automatically deployed every time you start an Oracle Event Processing server instance. You are not allowed to uninstall the dataservices application.

The purpose of this application is to provide a filter for diagnostic monitoring metrics. The application is itself an Oracle Event Processing application made up of adapters, streams, and a processor.

1.3.2 Manage Servers

Using Oracle Event Processing Visualizer, you can manage Oracle Event Processing server instances and the services they provide such as JMX, data sources, Jetty Web server, work managers, the persistent event store, event type repository, HTTP publish-subscribe server, and logs.

For more information, see Server and Domain Tasks.

1.3.3 Manage Security

Using Oracle Event Processing Visualizer, you can manage Oracle Event Processing server and application security including users, groups, and roles, SSL, and HTTP publish-subscribe server access.

For more information, see Security Tasks.

1.3.4 Update Configuration Data

Although you can update much of the configuration of an Oracle Event Processing server instance and its deployed applications, not all fields can be updated. The following rules determine what fields can be updated:

- Information in the EPN assembly file is static and thus read-only. Examples of this type of information include the stages of the EPN and how they are wired together.
- Information in the component configuration files can be modified, although not typically added to or deleted from. The next bullet lists the two exceptions. Examples of this type of information include the maximum size and threads of a stream.

  When you are allowed to update fields on a Oracle Event Processing Visualizer window, you will see three buttons: **Edit**, **Save**, and **Cancel**. Click the **Edit** button to modify the fields, then click **Save** to commit the changes to the server or **Cancel** to cancel.
- The Oracle CQL rules associated with a processor and the channels associated with an HTTP publish-subscribe server can be modified.
- Some information in the Oracle Event Processing server's configuration file (`config.xml`) can be modified, although much of it is read-only. An example of

this type of information includes the configuration of work managers, the logging service, and user-defined channels of the HTTP publish-subscribe server.

Note:

Do not modify or delete the internal channels of the HTTP publish-subscribe server. If you modify or delete these internal channels, Oracle Event Processing Visualizer will not function properly

Server configuration updates also use the three buttons: **Edit**, **Save**, and **Cancel**.

Note:

The preceding rules assume that you have logged onto Oracle Event Processing Visualizer with the required authentication credentials for performing the desired update task.

For more information, see *Administering Oracle Event Processing*.

1.4 Who Uses Oracle Event Processing Visualizer?

Oracle Event Processing Visualizer provides valuable services to a variety of Oracle Event Processing users, including:

- Administrators
- Developers
- Business Users.

1.4.1 Administrators

Administrators who use Oracle Event Processing Visualizer to connect to an Oracle Event Processing server instance use role-based authorization to gain access. Users that successfully authenticate themselves when using Oracle Event Processing Visualizer are assigned roles based on their group membership, and then subsequent access to administrative functions is restricted according to the roles held by the user. Anonymous users (non-authenticated users) will not have any access to Oracle Event Processing Visualizer.

When an administrator uses the Configuration Wizard to create a new domain, they enter an administrator user that will be part of the `wlevsAdministrators` group. By default, this information is stored in a file-based provider filestore. The password is hashed using the SHA-256 algorithm. Once the domain has been created, the administrator can create new groups using Oracle Event Processing Visualizer, assign roles to them, and then create new users and assign them to groups.

For more information, see User, Group, and Role Management.

Note:

The security features of Oracle Event Processing Visualizer work only if you have security enabled for Oracle Event Processing server. If you disable Oracle Event Processing server security, then:

- Oracle Event Processing Visualizer does not provide default users, groups, and roles.
- You cannot create new users, groups, and roles.
- There is still a login page when first entering Oracle Event Processing Visualizer but you may enter anything for user and password.
- Anonymous users may access the Oracle Event Processing Visualizer.

For more information, see *Administering Oracle Event Processing*

1.4.2 Developers

Developers can use Oracle Event Processing Visualizer to view server resources (such as data sources), perform event record and playback, and trouble shoot performance issues by turning on latency and throughput statistics.

1.4.3 Business Users

Business users can use the Oracle Event Processing Visualizer Query Wizard to create and modify queries within a given processor. This allows rule experts to manage Oracle CQL queries and views with minimal development assistance.

1.5 National Language Support

Oracle Event Processing Visualizer observes Java localization and supports the use of double-byte characters in all configuration files and Oracle Event Processing Visualizer text entry fields.

This section describes topics of interest when using Oracle Event Processing Visualizer with double-byte locales, including Configuration File Encoding: UTF-8.

Note:

By default, Oracle Event Processing Visualizer ships with an English resource bundle that supplies all the text that appears in the Oracle Event Processing Visualizer user interface.

1.6 Configuration File Encoding: UTF-8

Oracle Event Processing server encodes all configuration XML files using UTF-8 encoding. This encoding is specified in the header of all Oracle Event Processing configuration XML files. The following example shows the encoding specified in the `config.xml` file.

```
<?xml version="1.0" encoding="UTF-8"?>
<n1:config xsi:schemaLocation="http://www.bea.com/ns/wlevs/config/server
wlevs_server_config.xsd"
```

```
    xmlns:n1="http://www.bea.com/ns/wlevs/config/server"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <domain>
        <name>WLEventServerDomain</name>
    </domain>

    <netio>
        <name>NetIO</name>
        <port>9002</port>
    </netio>

    ...
</config>
```

When manually editing an Oracle Event Processing configuration XML file, be sure to save the file in UTF-8 encoding. Some editors will automatically save configuration XML files in the correct encoding based on the `encoding` attribute. However, some editors will not automatically save configuration XML files in the correct encoding. In this case, you must ensure that you select UTF-8 encoding when you save an Oracle Event Processing configuration XML file.

You can enter double-byte characters in any Oracle Event Processing Visualizer text field. The Oracle Event Processing Visualizer and Oracle Event Processing server will always write configuration XML files in the correct UTF-8 encoding.

Part I

Introduction

Introduction contains the following chapters:

- Oracle Event Processing Visualizer
- Use Visualizer.

2

Use Visualizer

This chapter describes how to start and set preferences for Oracle Event Processing Visualizer, a tool that can manage, configure, and monitor Oracle Event Processing server instances and the applications deployed to the server instances.

This chapter includes the following sections:

- Start Visualizer
- Manage User Preferences
- Improve Accessibility

For more information, see Oracle Event Processing Visualizer.

2.1 Start Visualizer

Before you start Oracle Event Processing Visualizer, ensure that your system meets the minimum prerequisites as Prerequisites describes.

Choose how you want to start Oracle Event Processing Visualizer:

- Start Visualizer from a Browser
- Start Visualizer in a Multiserver Domain

After you start Oracle Event Processing Visualizer, you are ready to begin using Oracle Event Processing Visualizer to manage, configure, and monitor Oracle Event Processing server instances and the applications deployed to the server instances. For information on typical tasks, see:

- Application Tasks
- Server and Domain Tasks

  Security Tasks .

2.1.1 Prerequisites

Before starting Oracle Event Processing Visualizer, the following prerequisites must be met:

- For best performance, install version 10,0,22,87 of the Adobe Flash Player.

  For instructions on testing the current version of Flash Player installed on your computer, see the Version test for Adobe Flash Player Web site: `http://kb.adobe.com/selfservice/viewContent.do?externalId=tn_15507`.

  For more information, see `http://www.adobe.com/products/flashplayer/`.

- You must configure your computer display for a resolution of 1024x768 or higher.
- Oracle recommends that you use Internet Explorer 7 or higher.
- For some Oracle Event Processing Visualizer tasks, you must log into Oracle Event Processing Visualizer using an Oracle Event Processing server account configured for the administrator role. For more information, see User, Group, and Role Management.

2.1.2 Start Visualizer from a Browser

You can start the Oracle Event Processing Visualizer using any browser that meets the minimum perquisites as Prerequisites describes.

Note:

Do not use an IPV6-only browser because most Visualizer features are not supported.

Start Visualizer from a Browser

1. Confirm that your browser meets the necessary requirements.

   See Prerequisites.

2. Invoke the following URL in your browser:

   ```
   http://host:port/wlevs
   ```

   where *host* refers to the name of the computer on which Oracle Event Processing is running and *port* refers to the Jetty NetIO port configured for the server (default value `9002`).

   The port number is configured in the `config.xml` file in the *`Oracle/Middleware/my_oep/user_projects/domains/DOMAIN_DIR/servername/`*`config` directory. The port number is the value of the `port` child element of the `netio` object configured for the Jetty server, as shown in the following example (only relevant parts shown):

   ```
   <netio>
       <name>NetIO</name>
       <port>9002</port>
   </netio>
   ...
   <jetty>
       <name>JettyServer</name>
       <network-io-name>NetIO</network-io-name>
       ...
   </jetty>
   ```

   For example, if your browser is running on the same computer as Oracle Event Processing and you are using the default port, invoke the following URL:

   ```
   http://localhost:9002/wlevs
   ```

   If you want to use HTTPS to connect to Oracle Event Processing Visualizer, specify the SSL port number. This is the port assigned to the `netio` element referenced by the `secure-network-io-name` Jetty element. The default value is `9003`.

For example, if you have the following configuration (only relevant parts shown):

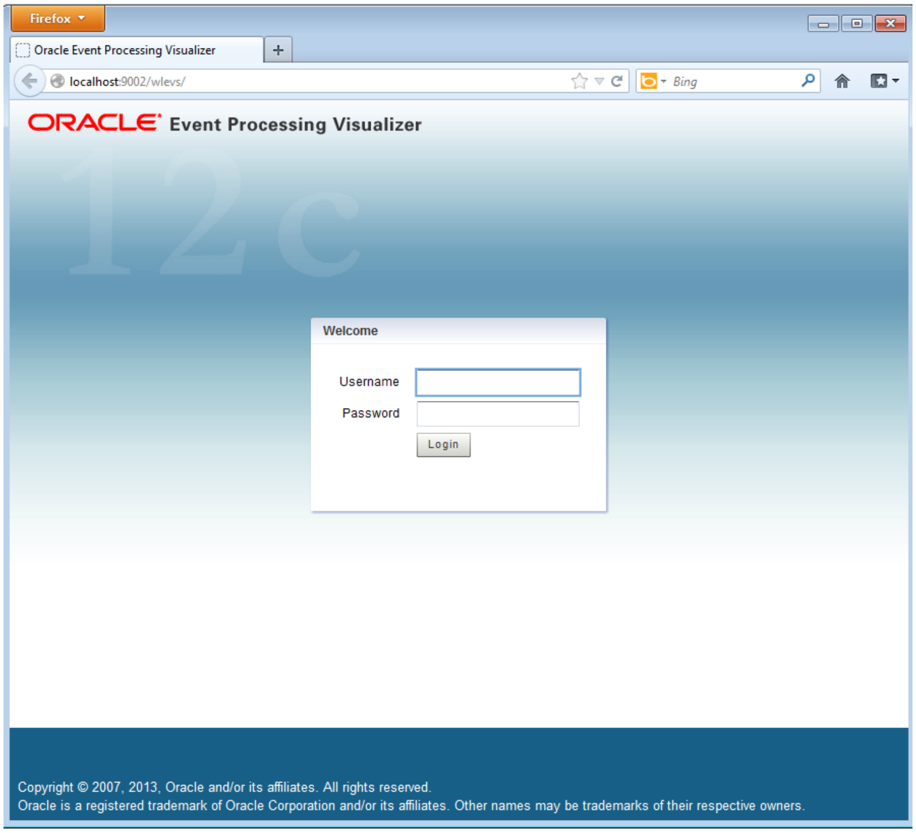
```
<netio>
    <name>sslNetIo</name>
    ...
    <port>9003</port>
</netio>
<jetty>
    ...
    <secure-network-io-name>sslNetIo</secure-network-io-name>
</jetty>
```

use the following URL:

```
https://localhost:9003/wlevs
```

The Login screen appears as Figure 2-1 shows.

Figure 2-1 Oracle Event Processing Visualizer Login Screen


Firefox
Oracle Event Processing Visualizer
localhost:9002/wlevs/
Bing
ORACLE Event Processing Visualizer
12c
Welcome
Username
Password
Login

Copyright © 2007, 2013, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

3. In the Login screen, enter the user name and password for the administrator user you configured when you created the domain.

Note:

The security features of Oracle Event Processing Visualizer work only if you have security enabled for Oracle Event Processing server. If you disable Oracle Event Processing server security, then:

- Oracle Event Processing Visualizer does not provide default users, groups, and roles.
- You cannot create new users, groups, and roles.
- There is still a login page when you first enter Oracle Event Processing Visualizer, but you can enter anything for the user name and password.

For more information, see *Administering Oracle Event Processing*

4. Click **Login**.

   Note that if you stop the server while Oracle Event Processing Visualizer is running, you might see informational messages about the service proxy. These are typically not errors.

2.1.3 Start Visualizer in a Multiserver Domain

You can use Oracle Event Processing Visualizer to administer a multiserver domain. As described in Start Visualizer from a Browser, Oracle Event Processing Visualizer works by connecting to one particular Oracle Event Processing server, based on its host and port.

In the case of a multiserver domain, you connect to one Oracle Event Processing server in the domain and then access the other servers from that server. All servers in a multiserver domain are candidates to host Oracle Event Processing Visualizer, from which you administer all the other servers in the domain. However, considering the performance factor, it is recommended that you select one server to host Oracle Event Processing Visualizer and disable Oracle Event Processing Visualizer access from all servers except for one in the multiserver domain.

To avoid a single point of failure, consider enabling Oracle Event Processing Visualizer on a small subset of `n` machines in the domain.

When using Oracle Event Processing Visualizer in a multiserver domain, the navigation tree in the left pane is automatically refreshed to reflect changes in the domain. For example, when a new server joins the domain, it automatically shows up in the navigation tree. Conversely, if the server leaves the domain, the server automatically disappears from the navigation tree.

Note:

The `server-host-name` element is mandatory if one or more Oracle Event Processing servers in your multiserver domain are on different hosts and you plan to manage the multiserver domain using the Oracle Event Processing Visualizer. It is also mandatory if a server is deployed on a host machine that has multiple IP addresses configured (whether in a multiserver or standalone-server environment). You must configure `cluster` element child element `server-host-name` in the `config.xml` of each Oracle Event Processing server.

For more information, see *Administering Oracle Event Processing*.

You can perform configuration management, of both the servers and applications, and operation management (such as diagnostics and event record and playback) on all servers in the multiserver domain.

Start Visualizer in a Multiserver Domain

1. Create a multiserver domain.

   For example, assume you have a multiserver domain with three servers (`defaultserver`, `server1`, and `server2`). Each server directory is a child directory of the domain directory, which is `/Oracle/Middleware/my_oep/user_projects/domains/myDomain`.

   For more information, *Administering Oracle Event Processing*.

2. Configure SSL to secure communication between the servers in the multiserver domain.

   For more information, see *Administering Oracle Event Processing*.

3. Choose one Oracle Event Processing server in the multiserver domain to host Oracle Event Processing Visualizer access.

   In this example, `defaultserver` will host Oracle Event Processing Visualizer access.

4. Specify the `-disablevisualizer` flag when you start the other servers in the multiserver domain that will not provide access to Oracle Event Processing Visualizer.

   You want the `defaultserver` to host Oracle Event Processing Visualizer and thus must disable access to Oracle Event Processing Visualizer from the other two servers. In this case, you would start each server as follows:

```
  prompt> cd /Oracle/Middleware/my_oep/user_projects/domains/myDomain/
defaultserver
  prompt> startwlevs
  prompt> cd ../server1
  prompt> startwlevs -disablevisualizer
  prompt> cd ../server2
  prompt> startwlevs -disablevisualizer
```

5. Configure SSL to secure communication between Oracle Event Processing Visualizer and defaultserver.

For more information, see Oracle Event Processing *Administering Oracle Event Processing*.

6. Start Oracle Event Processing Visualizer.

   For more information, see Start Visualizer from a Browser.

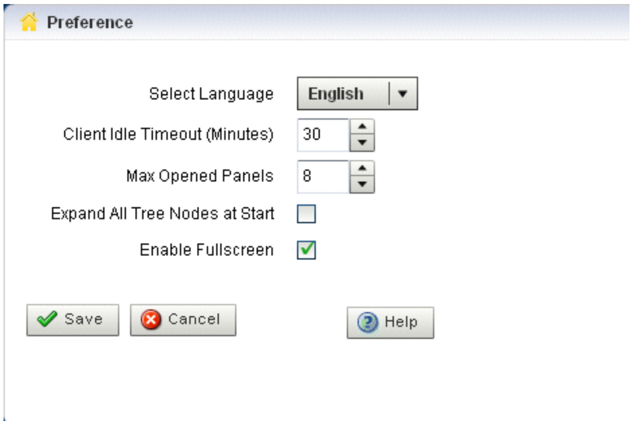
2.2 Manage User Preferences

Oracle Event Processing Visualizer allows you to customize its behavior using user preferences.

To manage user preferences:

1. Click the **Preference** button at the top-right corner of any Oracle Event Processing Visualizer screen. The Preference screen appears in the right panel, as shown in Figure 2-2.

Figure 2-2 The Preference Screen



Select Language: English

Client Idle Timeout (Minutes): 30

Max Opened Panels: 8

Expand All Tree Nodes at Start ☐

Enable Fullscreen ☑

2. Update the preferences as Table 2-1 describes:

Table 2-1 Oracle Event Processing Visualizer Preferences Attributes

Attribute	Description
`Select Language`	Select the language that Oracle Event Processing Visualizer uses. Default: English. For more information, see National Language Support.
`Client Idle Timeout: (Minutes)`	Set the time, in minutes, after which the client (browser) times out and automatically logs you out. Default value is 20 minutes, maximum 30.
`Max Opened Panels`	Set the maximum number of open items that will appear in the Open Items frame in the lower left corner. Default value is 8; maximum value is 10. **NOTE**: For users that require accessibility, set the maximum open items to 1 to make keyboard navigation easier to use. For more information, see Improve Accessibility.

Attribute	Description
`Expand All Tree Nodes at Start`	Click whether you want the domain tree in the left panel to be fully expanded every time you start Oracle Event Processing Visualizer
`Enable Fullscreen`	Click whether you want to enable the full screen function in Oracle Event Processing Visualizer. **NOTE**: For users that require accessibility, uncheck this item. For more information, see Improve Accessibility

3. Click **Save**.

   When the preferences have been successfully modified, a confirmation message appears momentarily.

2.3 Improve Accessibility

Oracle Event Processing Visualizer is a Web application that uses Adobe Flex. This section describes how to improve the accessibility of Oracle Event Processing Visualizer when using a screen reader like JAWS.

For more information, see:

- `http://www.adobe.com/accessibility/products/flex/jaws.html`

To improve Oracle Event Processing Visualizer accessibility:

1. Open the Oracle Event Processing Visualizer preferences panel.

   See Manage User Preferences.

2. Set the **Max Opened Panels** option to 1.

3. Uncheck the **Enable Fullscreen** option.

4. Click **Save**.

Part II

Application Tasks

Application Tasks contains the following chapters:

3

Application Tasks

This chapter introduces the typical application tasks you can perform with Oracle Event Processing Visualizer, including managing event processing networks (EPNs), application lifecycle, Oracle CQL rules, and configuration history.

This chapter includes the following sections:

- EPN Management
- Application Lifecycle Management
- Rules Management
- Configuration History Management.

3.1 EPN Management

The Oracle Event Processing Event Processing Network (EPN) represents the interconnections between the various Oracle Event Processing components of an Oracle Event Processing application. Using Oracle Event Processing Visualizer, you can manage the EPN, including viewing and changing the configuration of a stage, recording and playing back events, and monitoring the throughput and latency of a stage or path. Right-click a stage to select common stage-related functions from a convenient context menu.

For more information, see:

- Event Processing Network
- Event Processing Network Context Menu
- *Developing Applications for Oracle Event Processing*.

3.2 Application Lifecycle Management

You can manage the full application lifecycle including uploading an application, deploying it, suspending and resuming the application, and undeploying the application.

For more information, see:

- Application Lifecycle
- *Developing Applications for Oracle Event Processing*.

3.3 Rules Management

You can create, replace, and delete the rules in the Oracle CQL processors of Oracle Event Processing applications you develop and deploy as well as applications that Oracle Event Processing server deploys.

It is important to note that Oracle Event Processing Visualizer will update your configuration file on the server with any queries that you create. However, you will not see this take effect in your application from within the Oracle JDeveloper, in order to make the change there you will need to copy and paste the query from Oracle Event Processing Visualizer into your application.

For more information, see Oracle CQL Rules.

3.4 Configuration History Management

Using the Oracle Event Processing Visualizer, you can manage the configuration changes you make to Oracle Event Processing components. You can review change history and you can undo (roll back) changes to an earlier version. You can perform these operations on a given resource or application in both standalone-server and multiserver domains.

Alternatively, you can manage configuration history using the `wlevs.Admin` command-line tool.

For more information, see:

- Enable and Disable Configuration History Management
- Configure Change Record Purge Behavior
- Resource Configuration History Management
- Application Configuration History Management
- Application Lifecycle and Configuration History Management
- Multiserver Domains and Configuration History Management
- Configuration History
- *Administering Oracle Event Processing*.

3.4.1 Enable and Disable Configuration History Management

By default, Oracle Event Processing provides resource and application configuration history management.

Optionally, you can disable this feature by editing the startwlevs script for the server and adding the `com.oracle.ocep.config.version.enabled` system property to the last line (in practice, the command should be on one line):

```
"$JAVA_HOME/bin/java" $JVM_ARGS $DEBUG_ARGS -Dwlevs.home="$USER_INSTALL_DIR"
    -Dcom.oracle.ocep.config.version.enabled=false -Dbea.home="$BEA_HOME"
    -jar "${USER_INSTALL_DIR}/bin/wlevs.jar" $ARGS
```

If you disable this feature, Oracle Event Processing cannot propagate rules among the Oracle Event Processing servers in a multiserver domain. In this case, to propagate rule changes, you must manually update the rules in all the servers of a multiserver domain.

If you disable this feature and then wish to re-enable it, you must do so as follows:

1. Undeploy your Oracle Event Processing application.
2. Shut down the Oracle Event Processing server.
3. Edit the startwlevs script to either remove the `-Dcom.oracle.ocep.config.version.enabled` system property or set it to `true`.
4. Start the Oracle Event Processing server.
5. Deploy your Oracle Event Processing application.

   Failure to do so may cause rule propagation to fail in a multiserver domain and may prevent the Oracle Event Processing Visualizer from maintaining a consistent view of the Oracle Event Processing servers in a multiserver domain.

3.4.2 Configure Change Record Purge Behavior

You can set how often change records in the configuration history are purged, along with how many records are retained on each purge (in other words, records not purged). You might use these settings for enhancing performance. Purging more often and retaining fewer records might improve performance in some cases, such as when your application makes a large number of configuration changes dynamically at run time.

You configure these settings with system properties that you pass when starting the Oracle Event Processing server. The system properties are as follows:

- `com.oracle.ocep.config.version.changeRecordsPurgeFrequency` -- Amount of time, in seconds, between each purge. Default value is 3600 (one hour).
- `com.oracle.ocep.config.version.numSavedChangeRecords` -- Number of change records that will be saved when purging old change records. Default value is 25.

To make these settings, edit the startwlevs.cmd script for the server to add the system property to the last line (in actual practice, the command should be on one line).

For example, to set the purge frequency to 10 minutes and retain 50 records, you could use a command such as the following:

```
"$JAVA_HOME/bin/java" $JVM_ARGS $DEBUG_ARGS -Dwlevs.home="$USER_INSTALL_DIR"
    -Dcom.oracle.ocep.config.version.changeRecordsPurgeFrequency=600
    -Dcom.oracle.ocep.config.version.numSaveChangeRecords=50
    -Dbea.home="$BEA_HOME"
    -jar "${USER_INSTALL_DIR}/bin/wlevs.jar" $ARGS
```

You must restart the Oracle Event Processing server to have your changes take affect.

3.4.3 Resource Configuration History Management

You can manage the configuration history of the following Oracle Event Processing resources:

- Oracle Event Processing high availability adapters

- Oracle CQL rules

For more information, see Manage Resource Configuration History.

3.4.4 Application Configuration History Management

You can manage the configuration history for a given application. This allows you to see a summary of all the configuration changes made to all resources of a given application.

For more information, see Manage Application Configuration History.

3.4.5 Application Lifecycle and Configuration History Management

When you deploy an application, the Oracle Event Processing server creates a new configuration history for the application.

When you update an application, the Oracle Event Processing server erases the existing configuration history and creates a new configuration history for the application.

When you uninstall an application, the Oracle Event Processing server erases the configuration history for the application.

For more information, see Application Lifecycle Management.

3.4.6 Multiserver Domains and Configuration History Management

When you deploy an application to a multiserver domain, consider the following configuration history management restrictions:

- Rejoin a Multiserver Domain
- Simultaneous Updates

For more information, see Multiserver Domain Management.

3.4.6.1 Rejoin a Multiserver Domain

When an Oracle Event Processing server joins (or rejoins) a multiserver domain, the server's existing configuration history records (if any) are deleted and replaced with the current multiserver domain configuration history when the Oracle Event Processing server has synchronized with the primary.

3.4.6.2 Simultaneous Updates

Oracle Event Processing does not support simultaneous configuration updates from more than one Oracle Event Processing server in a multiserver domain.

Make configuration changes from one Oracle Event Processing Visualizer instance on one node in the multiserver domain.

4

Event Processing Network

This chapter describes how you can use Oracle Event Processing Visualizer to manage the event processing networks (EPNs) of a deployed Oracle Event Processing application. It explains how to view the EPN graphically, view and edit the configuration of a EPN stage, record and play back the event flow for debugging, tracing and injecting events. It also explains how to monitor channels and the throughput and latency of a stage or path in the EPN.

> **Note:**
>
> When one application (application-1) advertises a channel that a second application (application-2) subscribes to and uses as a foreign stage, you cannot view the details of the advertised channel from application-2. To view the details of this channel, you must do so from application-1.

This chapter includes the following sections:

- View the EPN of an Application
- View and Edit the Configuration of a Stage
- Record and Play Back Events in the EPN
- Trace and Inject Events in the EPN
- Monitor a Channel Stage in the EPN
- Monitor the Throughput and Latency of an EPN Stage or Path.

For more information, see EPN Management.

4.1 View the EPN of an Application

Using the Oracle Event Processing Visualizer, you can view the EPN of a deployed application.

View the EPN of an Application

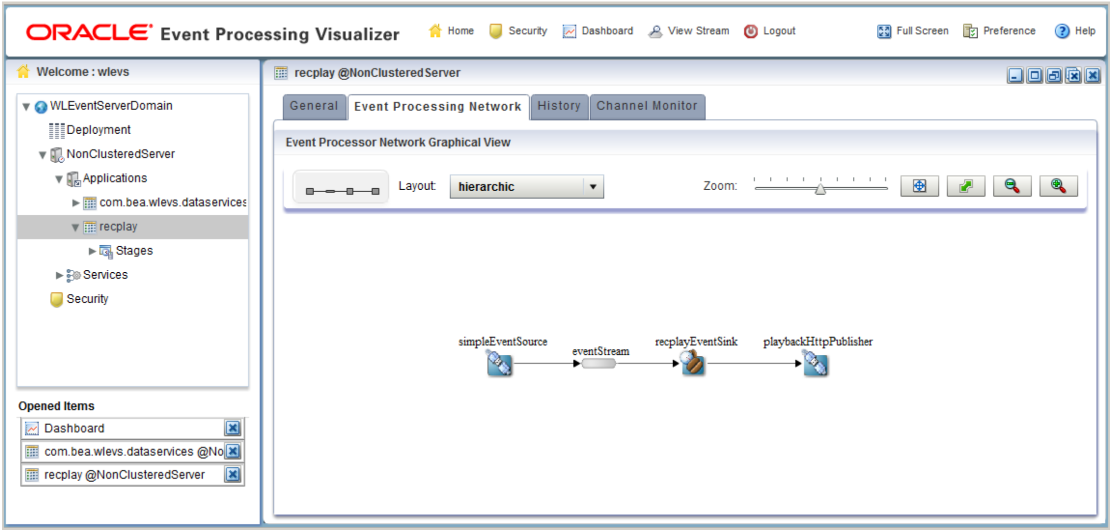
1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application whose EPN you want to view.

3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 4-1 shows.

Figure 4-1 Event Processing Network Panel



The name of a stage appears above the stage in black text. For example, there is an Oracle CQL processor called `FindCrossRates` and the name of its inbound channel is `FxQuoteStream`.

The name of the event type transmitted on a given channel is shown below the connecting line in blue text. For example, the event type transmitted on the `FxQuoteStream` is `CrossRateEvent`.

4. To navigate to areas of the EPN that are off the screen, click in the miniature EPN view, and drag.

5. To change the layout of the EPN, select a layout option from the Layout menu:

   - `organic`
   - `tree`
   - `hierarchic` (default)
   - `balloon`
   - `orthogonal`

6. To change the way the EPN fits in the browser window:

   a. Click on the **Zoom** slider and drag to the right to increase the zoom level; drag left to decrease the zoom level.

   b. Click the **Fit Content** button to automatically adjust the zoom level to make all of the EPN visible in the browser window; click the **Actual Size** button to reset the zoom level to zero.

   c. Click the **Zoom Out** (+) button or **Zoom In** (-) button.

4.2 View and Edit the Configuration of a Stage

Using Oracle Event Processing Visualizer, you can view the configuration of any stage and change the configuration for some stages.

Note:

Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multiserver domain, then these changes are not synchronized with the other servers in the same group. For more information, see Multiserver Domains.

4.2.1 How to View and Edit the Configuration of a Stage

You can view and change the configuration of a stage using the Oracle Event Processing Visualizer.

To view and edit the configuration of a stage:

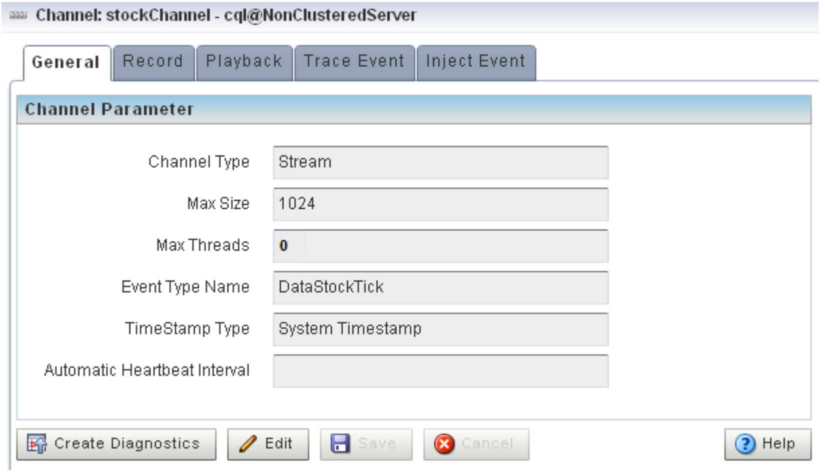
1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want.
3. Select the stage you wish to view and configure:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the stage you wish to view or Right-click the stage and select **Open Panel**.

   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to view.
      - Click the stage you wish to view.
      - In the right pane, click the **General** tab

   The stage's configuration appears.

   For example, Figure 4-2 shows the General tab for a channel.

Figure 4-2 General Tab for Channel Stage: Before Clicking Edit



Channel Type: Stream

Max Size: 1024

Max Threads: 0

Event Type Name: DataStockTick

TimeStamp Type: System Timestamp

Automatic Heartbeat Interval:

Note:

Not all stage's configuration can be updated by Oracle Event Processing Visualizer. An **Edit** button will appear if the configuration can be updated.

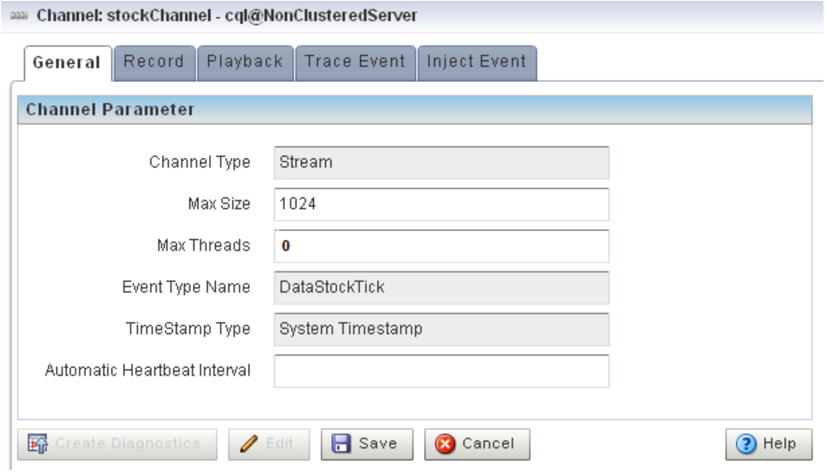
For a description of the configuration properties for each possible stage, see:

- Channel Properties
- Channel Properties: Outbound Channel
- Channel Properties: Outbound Channel With Query Selector
- Stream Properties
- Oracle CQL Processor Properties
- Adapter Properties
- JMS Adapter Properties
- High Availability Input Adapter Properties
- High Availability Buffering Output Adapter Properties
- High Availability Broadcast Output Adapter Properties
- High Availability Correlating Output Adapter Properties
- Event Bean Properties
- POJO Properties
- Cache Properties: Oracle Coherence Cache
- Cache Properties: Oracle Event Processing Local Cache.

4. Click the **Edit** button, if present.

   Modifiable attributes become editable as Figure 4-3 shows.

Figure 4-3 General Tab for Channel Stage: After Clicking Edit



Channel: stockChannel - cql@NonClusteredServer

Channel Type: Stream

Max Size: 1024

Max Threads: 0

Event Type Name: DataStockTick

TimeStamp Type: System Timestamp

Automatic Heartbeat Interval:

5. Enter new values for editable fields.

   For example, for the **Max Size** and **Max Threads** fields in Figure 4-3.

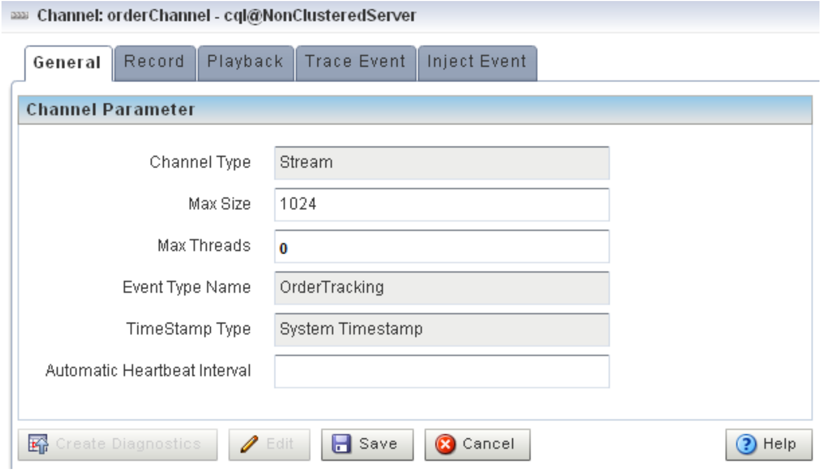
6. To commit your changes, click **Save**.

7. To leave the configuration unchanged, click **Cancel**.

4.2.1.1 Channel Properties

Figure 4-4 shows the General tab for a channel stage.

Figure 4-4 General Tab for Channel Stage



Channel: orderChannel - cql@NonClusteredServer

Channel Type: Stream

Max Size: 1024

Max Threads: 0

Event Type Name: OrderTracking

TimeStamp Type: System Timestamp

Automatic Heartbeat Interval:

After you click **Edit**, you can modify the attributes that are not shaded grey. Table 4-1 describes all the attributes on the General tab for this stage.

Table 4-1 General Tab Properties: Channel

Attribute	Description
`Channel Type`	Stream or Relation.

Attribute	Description
`Max Size`	Specifies the maximum size of the channel. Zero-size channels synchronously pass-through events. Non-zero size channels process events asynchronously, buffering events by the requested size. The default value is 0.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this channel. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 0.
`Event Type Name`	The name of the event type that this channel carries.
`Timestamp Type`	Use this element to specify whether or not the channel is application timestamped, that is, if the application is responsible for assigning a timestamp to each event, using any time domain. Valid values are: • `SYSTEM-TIME` (default) • `APPLICATION-TIME`
`Timestamp Property`	Use this element to specify an expression to be used as an application timestamp for event processing. Valid values are any Oracle CQL arithmetic expression
`Automatic HeartBeat`	For system timestamped relations or streams, time is dependent upon the arrival of data on the relation or stream data source. Oracle Event Processing generates a heartbeat on a system timestamped relation or stream if there is no activity (no data arriving on the stream or relation's source) for more than this number of nanoseconds. Either the relation or stream is populated by its specified source or Oracle Event Processing generates a heartbeat every `Automatic HeartBeat` number of nanoseconds. **Note**: This attribute is only applicable when a non-streaming source is connected to the channel.
`Is Total Order`	Indicates if application time published is always strictly greater than the last value used. Valid values are `true` or `false`. Default: `false`.

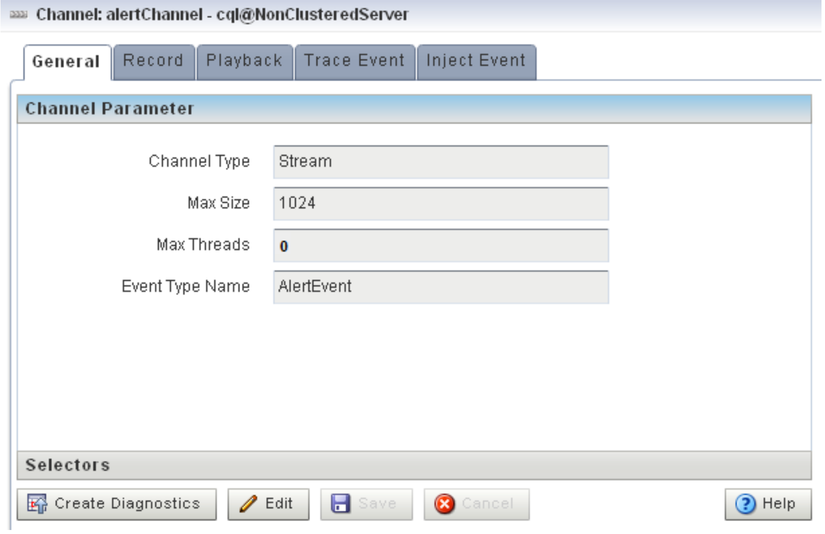
For more information, see:

- *Developing Applications for Oracle Event Processing*
- Channel Properties: Outbound Channel
- Channel Properties: Outbound Channel With Query Selector.

4.2.1.2 Channel Properties: Outbound Channel

Figure 4-5 shows the General tab for an outbound channel stage with an upstream Oracle CQL processor. The **Channel Parameter** accordion tab is selected.

Figure 4-5 General Tab Outbound Channel Parameters



Channel: alertChannel - cql@NonClusteredServer

Channel Parameter

Channel Type: Stream

Max Size: 1024

Max Threads: 0

Event Type Name: AlertEvent

After you click **Edit**, you can modify the attributes that are not shaded grey. Table 4-2 describes all the attributes on the General tab for this stage.

Table 4-2 General Tab Properties: Outbound Channel

Attribute	Description
`Channel Type`	Stream or Relation.
`Max Size`	Specifies the maximum size of the channel. Zero-size channels synchronously pass-through events. Non-zero size channels process events asynchronously, buffering events by the requested size. The default value is 0.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this channel. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 0.
`Event Type Name`	The name of the event type that this channel carries.

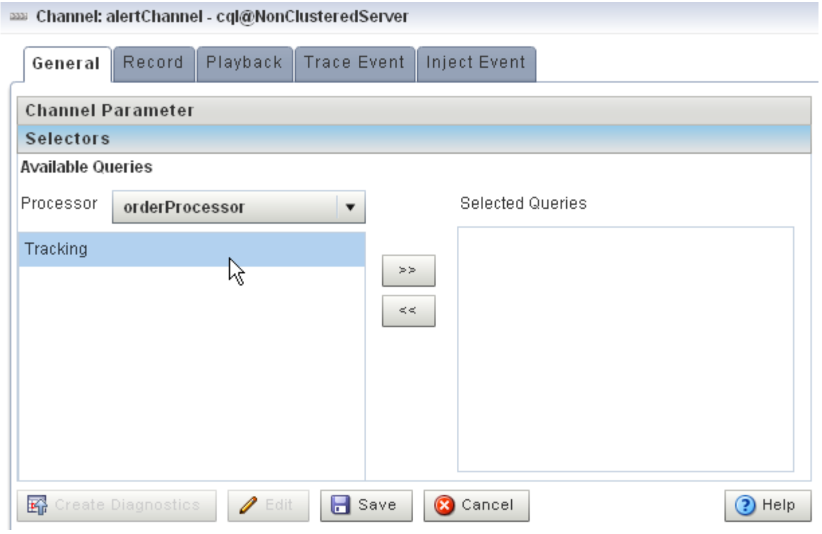
For more information, see:

- *Developing Applications for Oracle Event Processing*
- Channel Properties
- Channel Properties: Outbound Channel With Query Selector.

4.2.1.3 Channel Properties: Outbound Channel With Query Selector

Figure 4-6 shows the General tab for a channel stage with an upstream Oracle CQL processor. The **Selectors** accordion tab is selected.

Figure 4-6 General Tab Selectors for Outbound Channel Stage



Channel: alertChannel - cql@NonClusteredServer

Processor: orderProcessor

The Selector tab is only applicable if the up-stream node is an Oracle CQL processor.

After you click **Edit**, use the Selector tab, you can specify which up-stream Oracle CQL processor queries are permitted to output their results to the channel:

1. Select an up-stream Oracle CQL processor from the **Processor** pull-down menu.

   The available Oracle CQL rules associated with the selected processor are listed below the pull-down menu.

2. Use the left and right-pointing arrow buttons to move one or more rules to the **Selected Queries** list.

   Only rules in the Selected Queries list will output events to the channel.

> **Note:**
>
> You must add a query to the upstream Oracle CQL processor before you can add the query name to the channel selector. For more information, see Create a Rule with the Query Wizard.

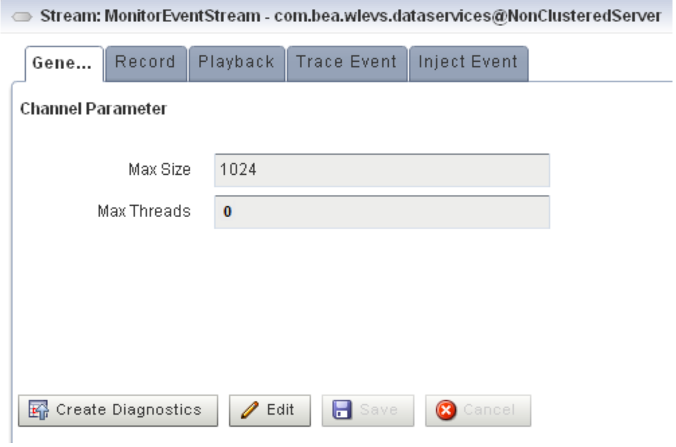
For more information, see:

- *Developing Applications for Oracle Event Processing*
- .
- Channel Properties: Outbound Channel .

4.2.1.4 Stream Properties

Figure 4-7 shows the General tab for a stream stage.

Figure 4-7 General Tab for Channel Stage



Channel Parameter

Max Size: 1024

Max Threads: 0

After you click **Edit**, you can modify the attributes that are not shaded grey. Table 4-3 describes all the attributes on the General tab for this stage.

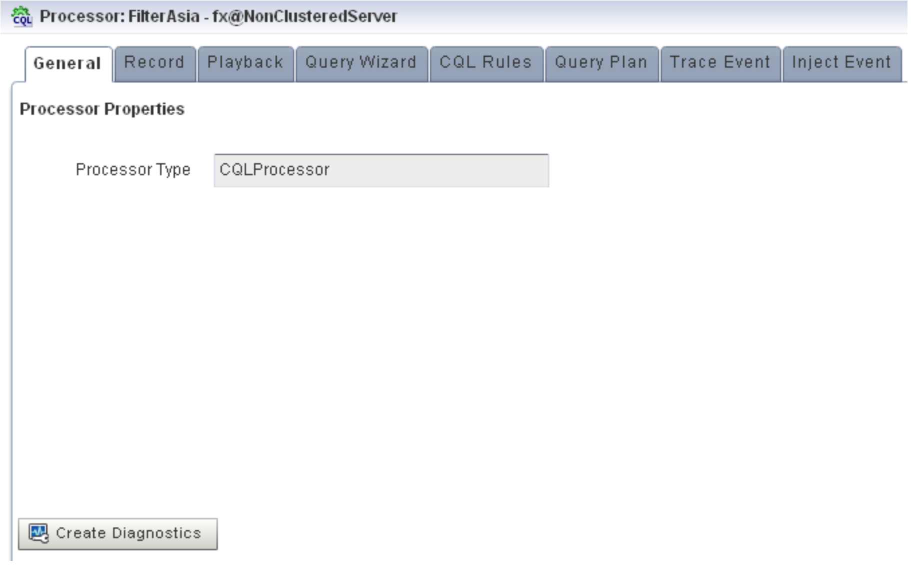
Table 4-3 General Tab Properties: Stream

Attribute	Description
`Max Size`	Specifies the maximum size of the stream. Zero-size streams synchronously pass-through events. Non-zero size streams process events asynchronously, buffering events by the requested size. The default value is 1024.
`Max Threads`	Specifies the maximum number of threads that will be used to process events for this stream. You can change `Max Threads` from 0 to a positive integer (that is, from a pass-through to multiple threads) without redeploying. However, if you change `Max Threads` from a positive integer to 0 (that is, from multiple threads to a pass-through), then you must redeploy your application. If the `Max Size` attribute is 0, then setting a value for `Max Threads` has no effect. The default value for this attribute is 1.

4.2.1.5 Oracle CQL Processor Properties

Figure 4-8 shows the General tab for an Oracle CQL processor stage.

Figure 4-8 General Tab for Oracle CQL Processor Stage



Processor Properties

Processor Type: CQLProcessor

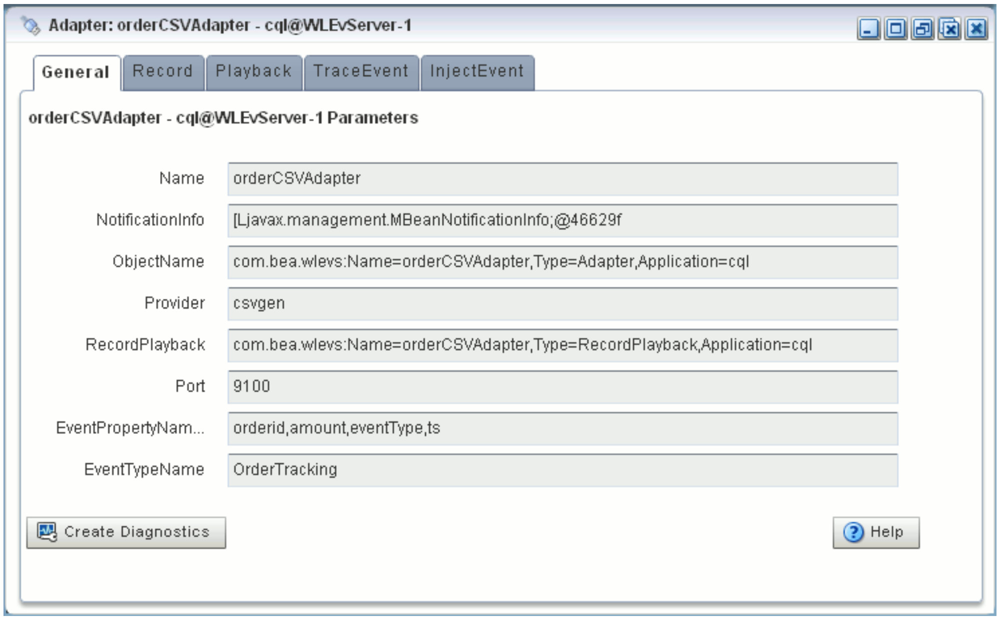
There are no editable properties for this type of stage.

See Oracle CQL*Developing Applications for Oracle Event Processing*.

4.2.1.6 Adapter Properties

Figure 4-9 shows the General tab for an adapter stage.

Figure 4-9 General Tab for an Adapter Stage



orderCSVAdapter - cql@WLEvServer-1 Parameters

Name	orderCSVAdapter
NotificationInfo	[Ljavax.management.MBeanNotificationInfo;@46629f
ObjectName	com.bea.wlevs:Name=orderCSVAdapter,Type=Adapter,Application=cql
Provider	csvgen
RecordPlayback	com.bea.wlevs:Name=orderCSVAdapter,Type=RecordPlayback,Application=cql
Port	9100
EventPropertyNam...	orderid,amount,eventType,ts
EventTypeName	OrderTracking

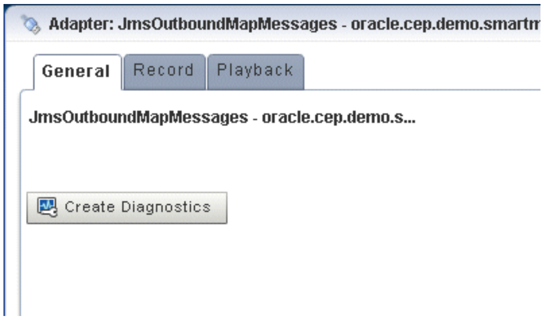
There are no editable properties for this type of stage.

Developing Applications for Oracle Event Processing.

4.2.1.7 JMS Adapter Properties

Figure 4-10 shows the General tab for a JMS adapter.

Figure 4-10 General Tab for a JMS Adapter Stage



Adapter: JmsOutboundMapMessages - oracle.cep.demo.smartm

JmsOutboundMapMessages - oracle.cep.demo.s...

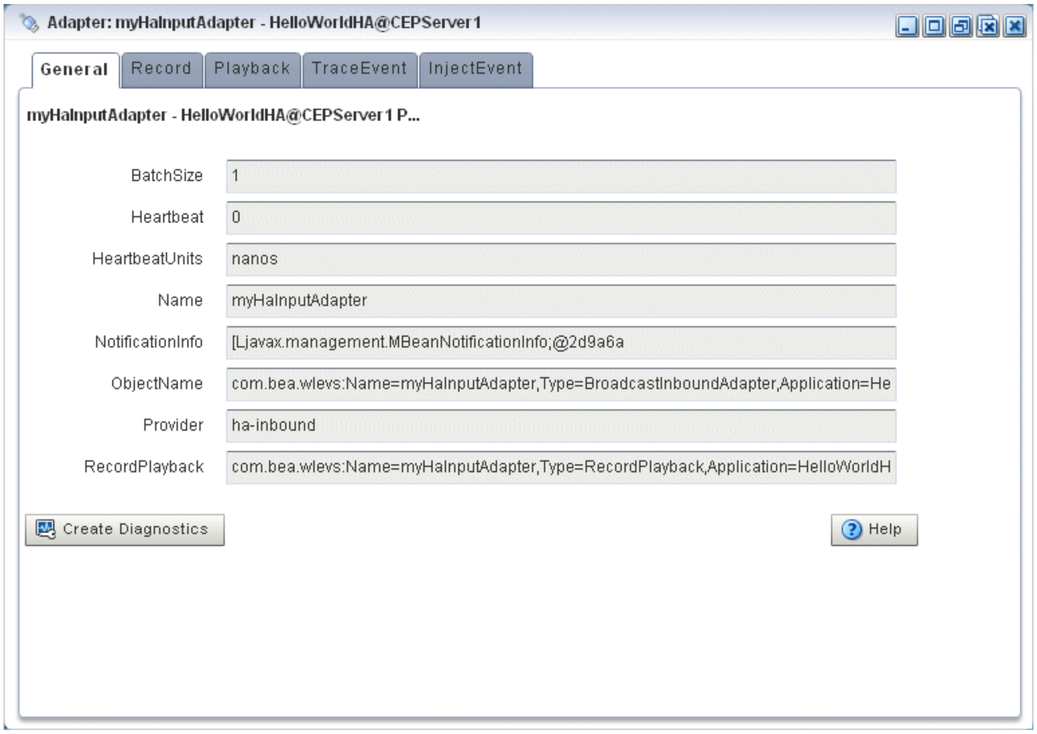
There are no editable properties for this type of stage.

Developing Applications for Oracle Event Processing.

4.2.1.8 High Availability Input Adapter Properties

Figure 4-11 shows the General tab for an Oracle Event Processing high availability input adapter.

Figure 4-11 General Tab for an Oracle Event Processing High Availability Input Adapter Stage



Adapter: myHaInputAdapter - HelloWorldHA@CEPServer1

myHaInputAdapter - HelloWorldHA@CEPServer1 P...

BatchSize	1
Heartbeat	0
HeartbeatUnits	nanos
Name	myHaInputAdapter
NotificationInfo	[Ljavax.management.MBeanNotificationInfo;@2d9a6a
ObjectName	com.bea.wlevs:Name=myHaInputAdapter,Type=BroadcastInboundAdapter,Application=He
Provider	ha-inbound
RecordPlayback	com.bea.wlevs:Name=myHaInputAdapter,Type=RecordPlayback,Application=HelloWorldH

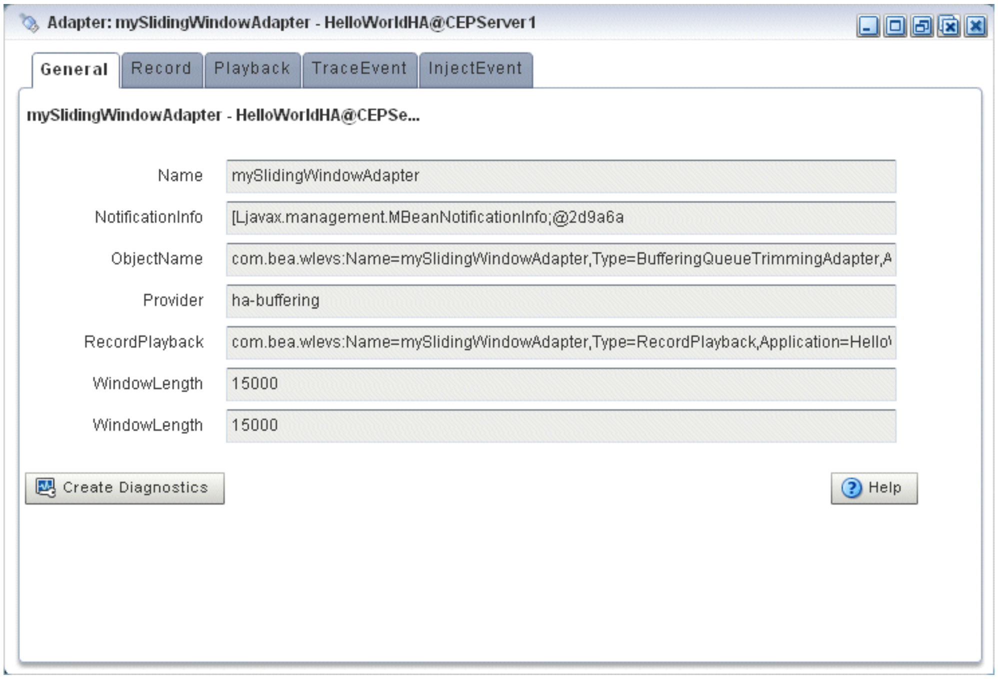
There are no editable properties for this type of stage.

See *Developing Applications for Oracle Event Processing*.

4.2.1.9 High Availability Buffering Output Adapter Properties

Figure 4-12 shows the General tab for an Oracle Event Processing high availability buffering output adapter.

Figure 4-12 General Tab for an Oracle Event Processing High Availability Buffering Output Adapter Stage


Adapter: mySlidingWindowAdapter - HelloWorldHA@CEPServer1
General
Record
Playback
TraceEvent
InjectEvent
mySlidingWindowAdapter - HelloWorldHA@CEPSe...
Create Diagnostics
Help

Name: mySlidingWindowAdapter

NotificationInfo: [Ljavax.management.MBeanNotificationInfo;@2d9a6a

ObjectName: com.bea.wlevs:Name=mySlidingWindowAdapter,Type=BufferingQueueTrimmingAdapter,A

Provider: ha-buffering

RecordPlayback: com.bea.wlevs:Name=mySlidingWindowAdapter,Type=RecordPlayback,Application=Hello\

WindowLength: 15000

WindowLength: 15000

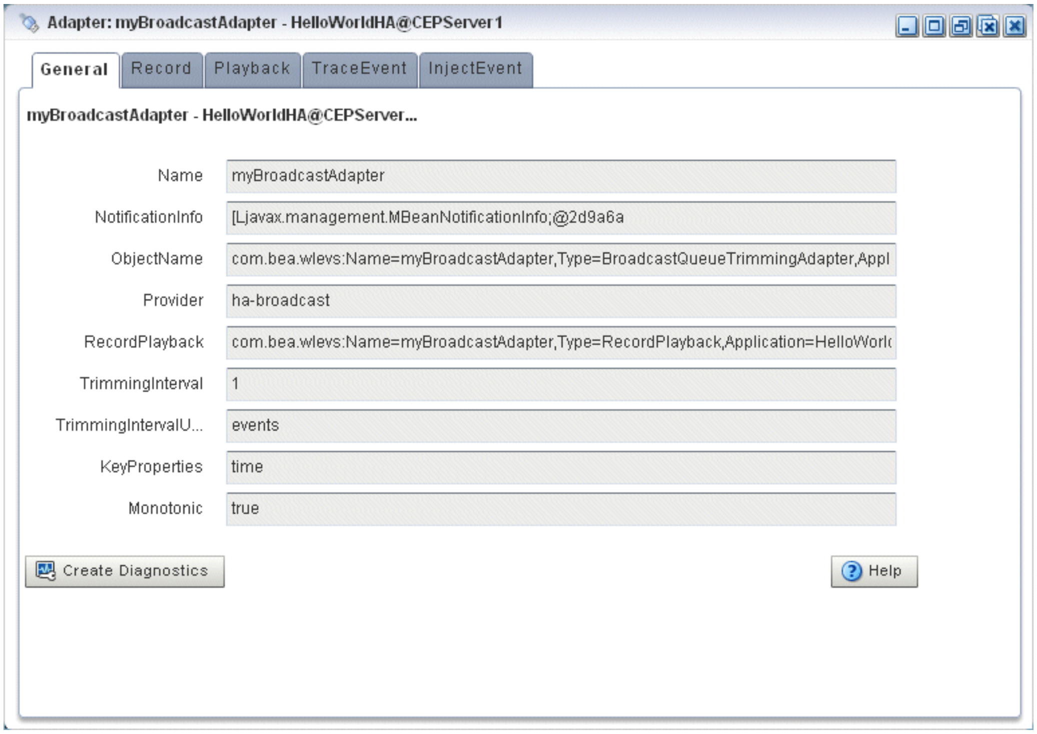
There are no editable properties for this type of stage.

See *Developing Applications for Oracle Event Processing*.

4.2.1.10 High Availability Broadcast Output Adapter Properties

Figure 4-13 shows the General tab for an Oracle Event Processing high availability broadcast output adapter.

Figure 4-13 General Tab for a High Availability Broadcast Output Adapter Stage



myBroadcastAdapter - HelloWorldHA@CEPServer...

Name	myBroadcastAdapter
NotificationInfo	[Ljavax.management.MBeanNotificationInfo;@2d9a6a
ObjectName	com.bea.wlevs:Name=myBroadcastAdapter,Type=BroadcastQueueTrimmingAdapter,Appl
Provider	ha-broadcast
RecordPlayback	com.bea.wlevs:Name=myBroadcastAdapter,Type=RecordPlayback,Application=HelloWorl(
TrimmingInterval	1
TrimmingIntervalU...	events
KeyProperties	time
Monotonic	true

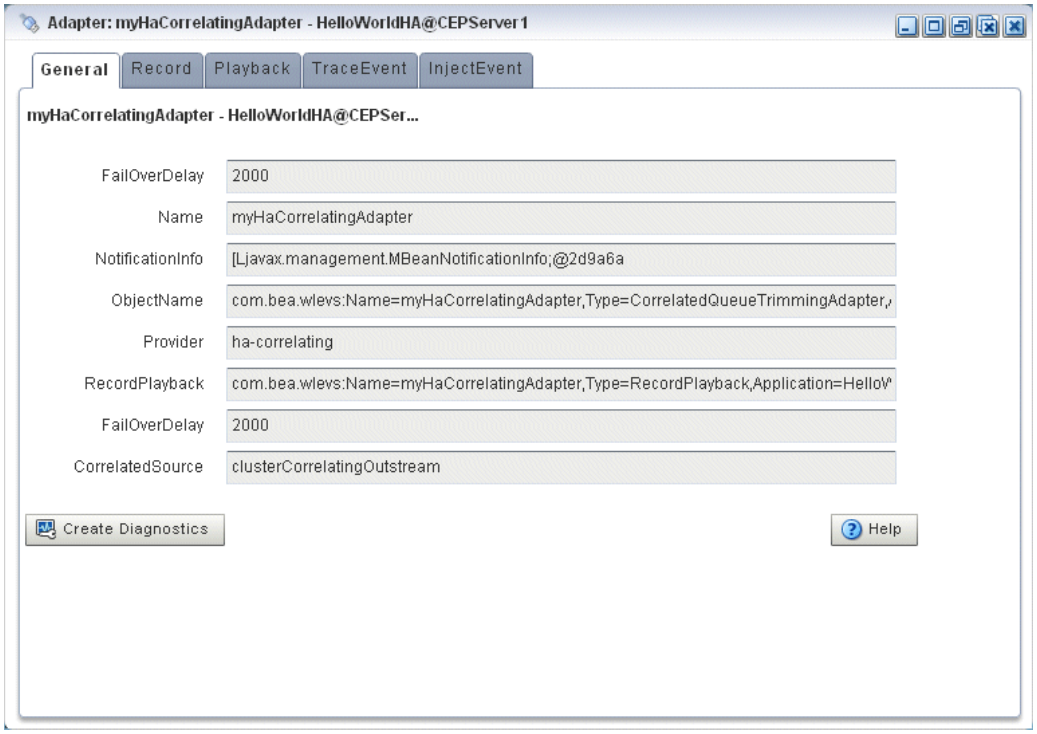
There are no editable properties for this type of stage.

See *Developing Applications for Oracle Event Processing*.

4.2.1.11 High Availability Correlating Output Adapter Properties

Figure 4-14 shows the General tab for an Oracle Event Processing high availability correlating output adapter.

Figure 4-14 General Tab for an Oracle Event Processing High Availability Correlating Output Adapter Stage



myHaCorrelatingAdapter - HelloWorldHA@CEPSer...

FailOverDelay: 2000
Name: myHaCorrelatingAdapter
NotificationInfo: [Ljavax.management.MBeanNotificationInfo;@2d9a6a
ObjectName: com.bea.wlevs:Name=myHaCorrelatingAdapter,Type=CorrelatedQueueTrimmingAdapter,
Provider: ha-correlating
RecordPlayback: com.bea.wlevs:Name=myHaCorrelatingAdapter,Type=RecordPlayback,Application=HelloW
FailOverDelay: 2000
CorrelatedSource: clusterCorrelatingOutstream

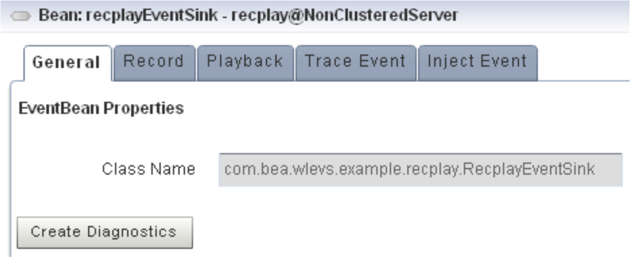
There are no editable properties for this type of stage.

See *Developing Applications for Oracle Event Processing*.

4.2.1.12 Event Bean Properties

Figure 4-15 shows the General tab for an event bean stage.

Figure 4-15 General Tab for Event Bean Stage



EventBean Properties

Class Name: com.bea.wlevs.example.recplay.RecplayEventSink

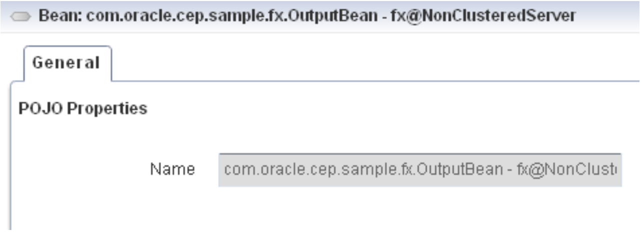
There are no editable properties for this type of stage.

See *Developing Applications for Oracle Event Processing*.

4.2.1.13 POJO Properties

Figure 4-16 shows the General tab for a Plain Old Java Object (POJO) stage.

Figure 4-16 General Tab for POJO Stage



Bean: com.oracle.cep.sample.fx.OutputBean - fx@NonClusteredServer

POJO Properties

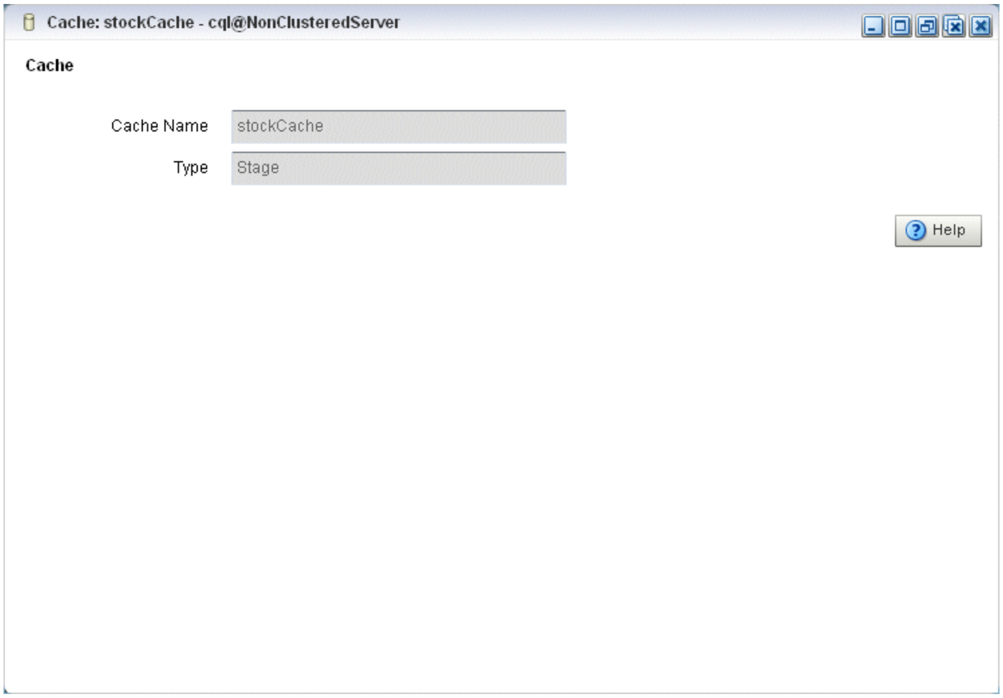
Name: com.oracle.cep.sample.fx.OutputBean - fx@NonClust

There are no editable properties for this type of stage.

4.2.1.14 Cache Properties: Oracle Coherence Cache

Figure 4-17 shows the properties for an Oracle Coherence Cache stage.

Figure 4-17 Oracle Coherence Cache Stage Properties



Cache: stockCache - cql@NonClusteredServer

Cache

Cache Name: stockCache

Type: Stage

There are no editable properties for this type of stage.

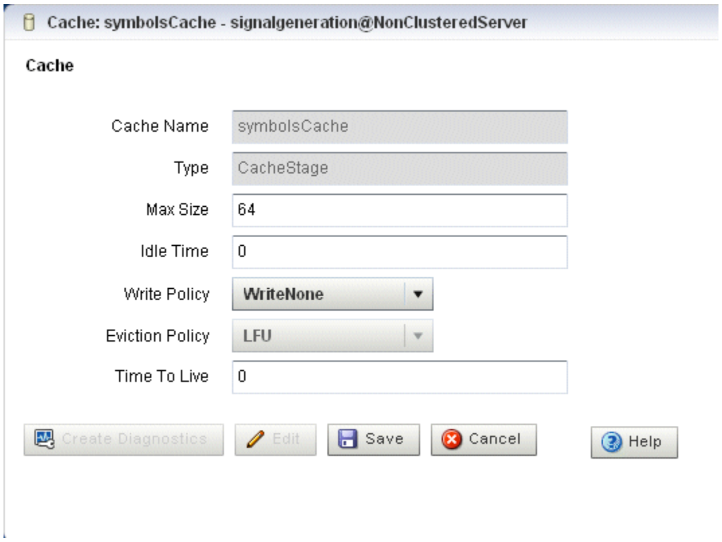
For more information, see:

- Multiserver Domains
- See *Developing Applications for Oracle Event Processing*.

4.2.1.15 Cache Properties: Oracle Event Processing Local Cache

Figure 4-18 shows the properties for an Oracle Event Processing Local Cache stage.

Figure 4-18 Oracle Event Processing Local Cache Stage Properties



Cache: symbolsCache - signalgeneration@NonClusteredServer

Cache

Cache Name: symbolsCache

Type: CacheStage

Max Size: 64

Idle Time: 0

Write Policy: WriteNone

Eviction Policy: LFU

Time To Live: 0

After you click **Edit**, you can modify the attributes that are not shaded grey. Table 4-4 describes all the attributes on the General tab for this stage.

Table 4-4 Properties: Oracle Event Processing Local Cache

Attribute	Description
`Max Size`	Specifies the `max-size` element to define the number of cache elements in memory after which eviction or paging occurs. Currently, the maximum cache size is 2^31-1 entries. This element may be changed dynamically.
`Idle Time`	Specifies the number of milliseconds a cache entry may not be accessed before being actively removed from the cache. By default, there is no `idle-time` set. This element may be changed dynamically.
`Write Policy`	Specifies how Oracle Event Processing server writes information into the cache. Valid values are: • `WriteNone`: Specifies no writes to a cache store. This is the default write policy. This element can be changed dynamically. • `WriteThrough`: Specifies synchronous writes to the cache store. As soon as an entry is created or updated the write occurs. This element may be changed dynamically. • `WriteBehind`: Specifies asynchronous writes to the cache store. The cache store is invoked from a separate thread after a create or update of a cache entry. This element may be changed dynamically.
`Eviction Policy`	Use this element to define the eviction policy the cache uses when `Max Size` is reached. Valid values are: • `FIFO`: first in, first out. • `LRU`: least recently used • `LFU`: least frequently used (default) • `NRU`: not recently used
`Time To Live`	Specifies the maximum amount of time, in milliseconds, that an entry is cached. Default value is 0 (which means infinite).

4.3 Record and Play Back Events in the EPN

The event repository feature of Oracle Event Processing allows you to record events flowing through an event processing network (EPN) and store them so you can later play back the events. You configure the recording and playing back of events per stage, such as a processor or stream. Additionally, only events coming out of an event source can be recorded, and playback is possible only on event sinks (events are played back to the inbound side of the event sink stage.)

The only configuration options of record and playback that you can control using Oracle Event Processing Visualizer are event type, time, and speed.

For detailed information about how event and record playback works and how to configure a component, see *Developing Applications for Oracle Event Processing*.

Alternatively, you can trace and inject events as Trace and Inject Events in the EPN describes.

This section describes:

- How to Record Events
- Play Back Events.

4.3.1 How to Record Events

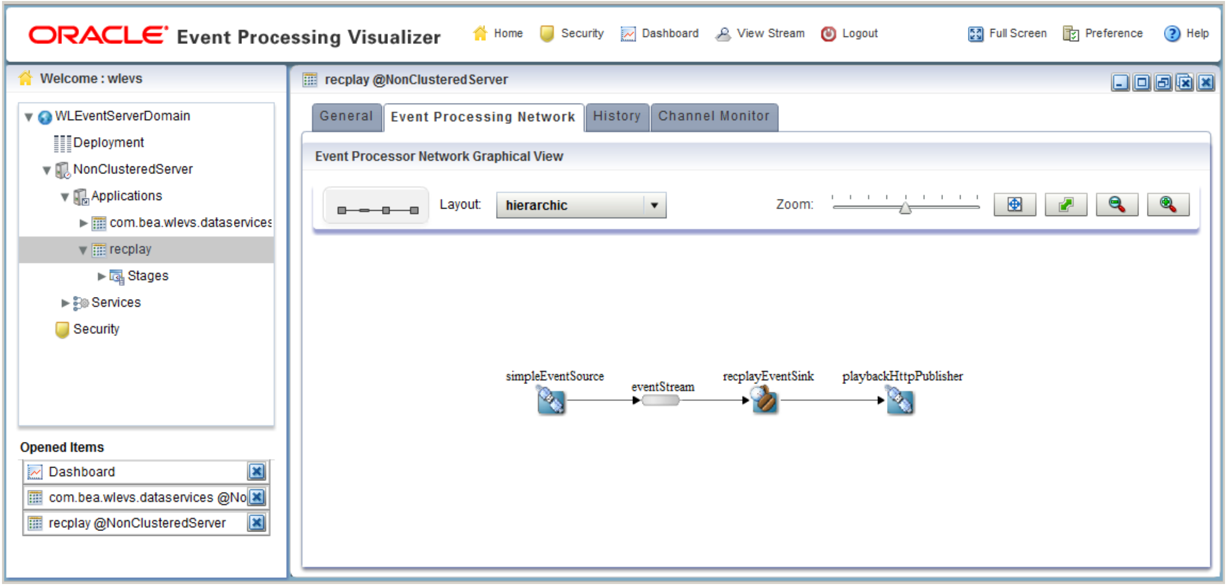
Using Oracle Event Processing Visualizer you can record events for a selected stage. Later, you can playback these events (see Play Back Events).

To record events:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.
3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 4-19 shows.

Figure 4-19 Event Processing Network Panel



4. Select the stage for which you wish to record an event:

   a. To use the EPN diagram:

      - Right-click the stage for which you wish to record an event and select **Record Event**.

   b. To use the domain tree:

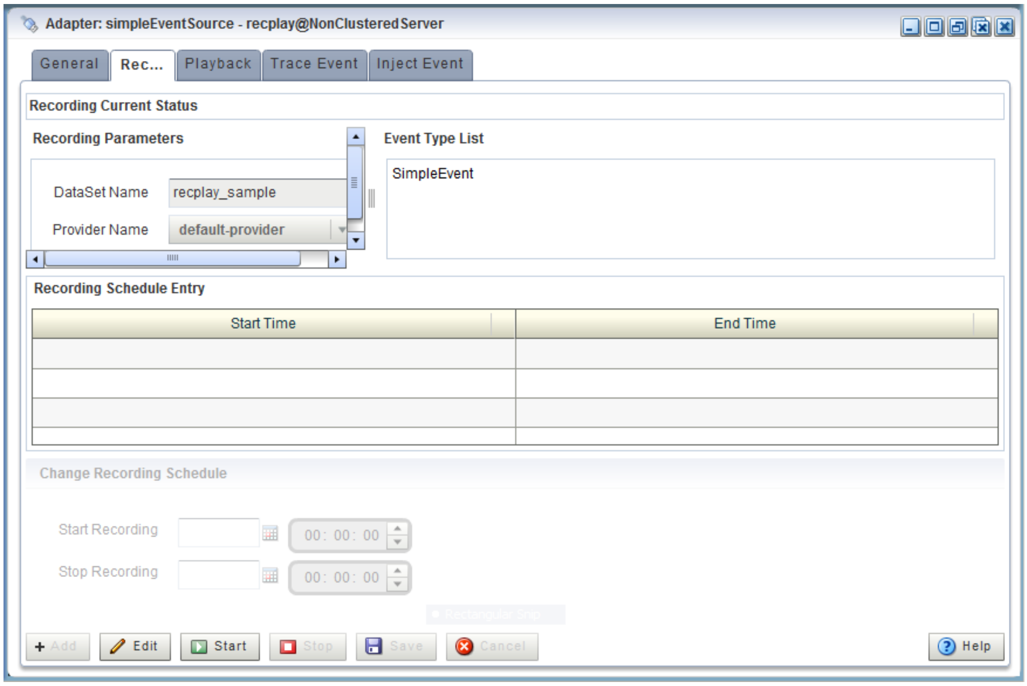
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.

      - Click the stage for which you wish to record an event.

      - In the right pane, click the **Record** tab

   The Record panel appears as shown in Figure 4-20.

Figure 4-20 The Record Tab


Adapter: simpleEventSource - recplay@NonClusteredServer

DataSet Name: recplay_sample

Provider Name: default-provider

Event Type List: SimpleEvent

Start Time	End Time

Start Recording: 00 : 00 : 00

Stop Recording: 00 : 00 : 00

The Record tab in Oracle Event Processing Visualizer for a particular stage is divided into the following sections:

- **Recording Current Status**: displays the current status of a recording. When the system has begun a recording session, then this field displays a blinking Recording message and it changes back to blank when the recording sessions ends. This section is read-only.
- **Recording Parameters**: specifies the name of the database schema (Dataset name) and the provider information. You must pre-configure the provider for the event repository.

  For more information, see:

  – See *Developing Applications for Oracle Event Processing*

  – Persistent Event Store.
- **Event Type List**: contains the Event Type List pane that displays the list of event types that are associated with a selected stage of the Oracle Event Processing application. Choose one or more event types to record.
- **Record Schedule Entry**: entries in this table display the recording start time and end time. The start and end time entries are optional fields. To start recording immediately, click the **Start** button at the bottom of the panel.
- **Change Recording Schedule**: allows you to specify and change the start time and end time of the recording using the calendar and clock controls.

5. Decide what you want to do:

   a. To add a new record entry, click **Add**.

   b. To modify an existing record, click **Edit**.

c. To delete an existing record schedule entry:

In the **Recording Schedule Entry** table, select the entry and click **Edit**.

Under **Change Recording Schedule**, click **Clear Schedule** button.

Click **Save**.

6. Configure the **Recording Parameters**:

- DataSet Name: the value of the record-parameters child element dataset-name element from the simpleEventSource adapter application configuration file /Oracle/Middleware/my_oep/oep/examples/domains/recplay_domain/defaultserver/applications/recplay/config.xml.

```
<adapter>
    <name>simpleEventSource</name>
    <record-parameters>
        <dataset-name>recplay_sample</dataset-name>
        <event-type-list>
            <event-type>SimpleEvent</event-type>
        </event-type-list>
        <provider-name>test-rdbms-provider</provider-name>
        <batch-size>1</batch-size>
        <batch-time-out>10</batch-time-out>
    </record-parameters>
</adapter>
```

- Provider Name: the value of the rdbms-event-store-provider child element name which corresponds to the data-source child element.

  When using the default Berkeley database provider, select default provider.

```
<data-source>
    <name>derby1</name>
    <connection-pool-params>
        <initial-capacity>15</initial-capacity>
        <max-capacity>50</max-capacity>
    </connection-pool-params>
    <driver-params>
        <url>jdbc:derby:dbtest1;create=true</url>
        <driver-name>org.apache.derby.jdbc.EmbeddedDriver</driver-name>
    </driver-params>
</data-source>
<rdbms-event-store-provider>
    <name>test-rdbms-provider</name>
    <data-source-name>derby1</data-source-name>
</rdbms-event-store-provider>
```

  For more information, see Data Source Configuration.

7. Configure the **Event type list**:

   Use the left and right-pointing arrow buttons to move the events you want recorded from the **Available event list** to the **Recorded event list**.

   For more information, see Event Type Repository.

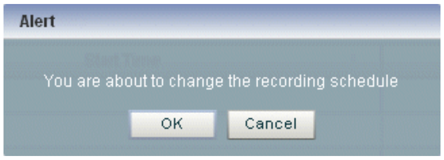
8. Decide when you want recording to start and end:

   a. If you want to schedule a record start and end time:

- Click the calendar and clock controls to add a start time and end time.
- Click **Save**.

  An alert dialog appears as Figure 4-21 shows.

  Figure 4-21 Record Schedule Confirmation Dialog

  Alert

  You are about to change the recording schedule

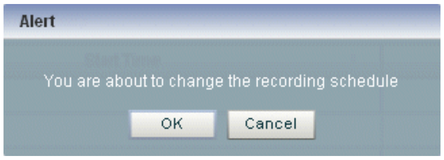
- Click **OK**.

b. If you want to record events immediately:

- Click **Save**.

  An alert dialog appears as Figure 4-22 shows.

  Figure 4-22 Record Schedule Confirmation Dialog

  Alert

  You are about to change the recording schedule

- Click **OK**.
- Click **Start**.

Oracle Event Processing Visualizer keeps track of whether a particular stage is currently recording or playing back events; based on this information, the Start and End buttons may be enabled or disabled as appropriate.

When you start recording using Oracle Event Processing Visualizer, Oracle Event Processing uses the event type information from your record entry and begins recording immediately; the pre-scheduled time, if any, remains unchanged. Use the End button to stop recording of your session immediately.

9. Decide when you want to stop recording:

   a. If you scheduled a stop time, event recording will stop at that time.

   b. If you want to stop event recording immediately, click **Stop**.

4.3.2 Play Back Events

Using Oracle Event Processing Visualizer you can playback previously recorded events for a selected stage.

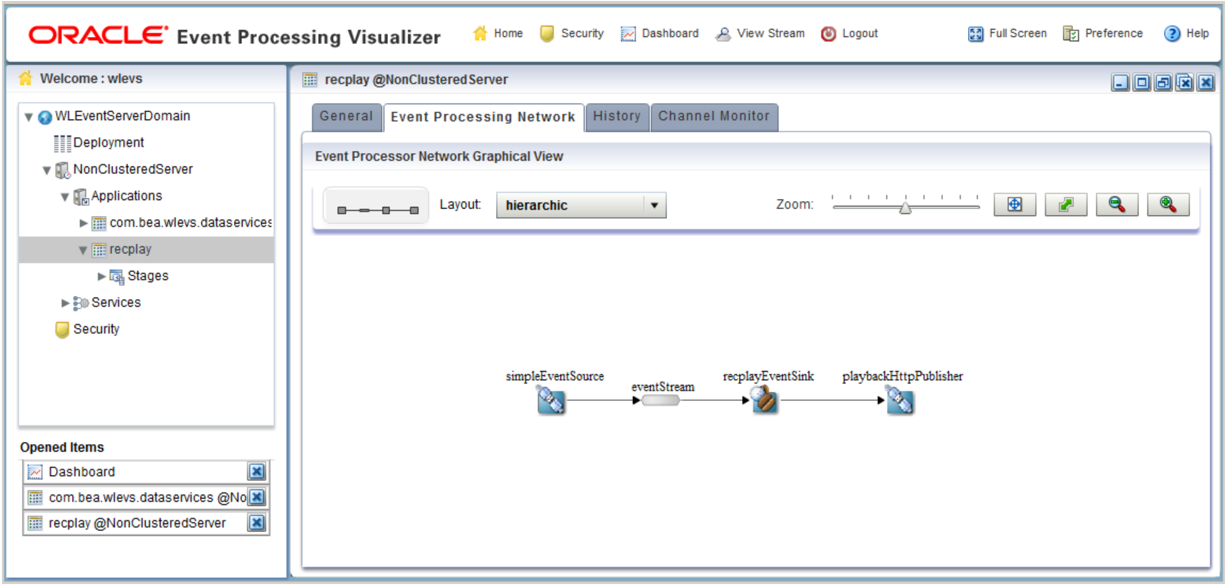
Play Back Events

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to record and playback events with.

3. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 4-23 shows.

Figure 4-23 Event Processing Network Panel



Layout: hierarchic

4. Select the stage for which you wish to playback an event:

   a. To use the EPN diagram:

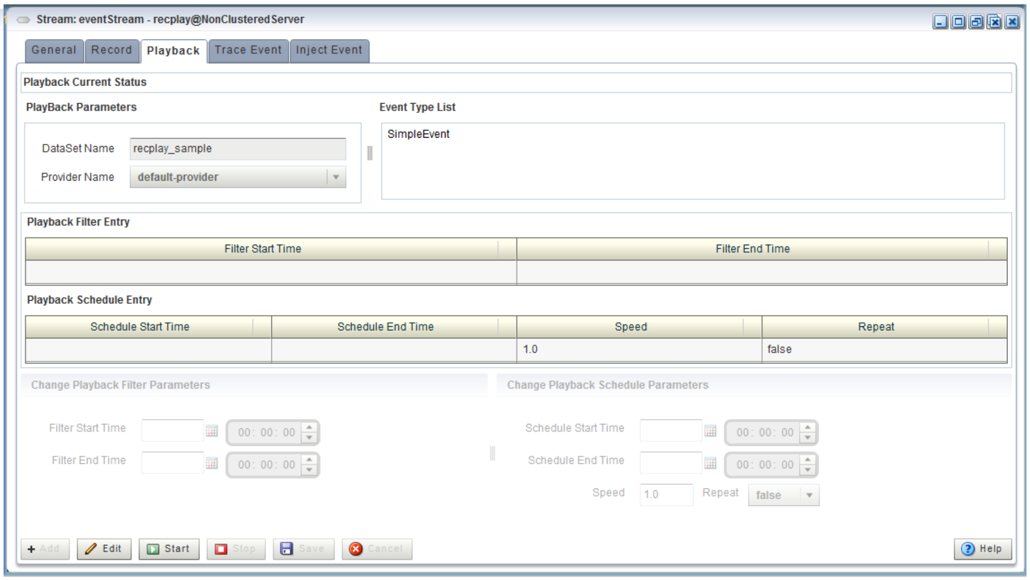
      - Right-click the stage for which you wish to playback an event and select **Playback Event**.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to record and playback events with.
      - Click the stage for which you wish to playback an event.
      - In the right pane, click the **Playback** tab

   The Record panel appears as shown in Figure 4-24.

Figure 4-24 The Playback Tab



Playback Current Status

PlayBack Parameters

DataSet Name: recplay_sample

Provider Name: default-provider

Event Type List

SimpleEvent

Playback Filter Entry

Filter Start Time	Filter End Time

Playback Schedule Entry

Schedule Start Time	Schedule End Time	Speed	Repeat
		1.0	false

Change Playback Filter Parameters

Filter Start Time: 00 : 00 : 00

Filter End Time: 00 : 00 : 00

Change Playback Schedule Parameters

Schedule Start Time: 00 : 00 : 00

Schedule End Time: 00 : 00 : 00

Speed: 1.0 Repeat: false

The Playback tab in Oracle Event Processing Visualizer for a particular stage is divided into the following sections:

- **Playback Current Status**: displays the current status of playback. When the system has begun playing back a session, then this field displays a blinking Playback message and it changes back to blank when the playback session ends. This section is read-only.
- **Playback Parameters**: specifies the name of the database schema (Dataset name) and the provider information. You must pre-configure the provider for the event repository.

  For more information, see:

  – See *Developing Applications for Oracle Event Processing*

  – Persistent Event Store.
- **Event Type List**: contains the Event Type List pane that displays the list of event types that are associated with a selected stage of the Oracle Event Processing application. Choose one or more event types to playback.
- **Playback Filter Entry**: specify a set of filters using the clock and time controls, to run a playback for an event type in event type repository. Only events that were recorded during the start and end times you specify here will be played back.
- **Playback Schedule Entry**: entries in this table display the playback start time and end time. The start and end time entries are optional fields. To start a playback immediately, click the **Start** button at the bottom of the panel.
- **Change Playback Filter Parameters**: allows you to change some of the properties of the selected playback filter entry.
- **Change Playback Schedule Parameters**: allows you to change some of the properties of the selected playback schedule entry. Use the calendar and clock

controls to specify or change an existing start and end time for playing back events.

5. Configure the **Playback Parameters**:

   - `DataSet Name`: the value of the `record-parameters` child element `dataset-name` element from the `SimpleEventSource` adapter application configuration file `/Oracle/Middleware/my_oep/oep/examples/domains/recplay_domain/defaultserver/applications/recplay/config.xml`.

     ```
     <stream>
         <name>eventStream</name>
         <playback-parameters>
             <dataset-name>recplay_sample</dataset-name>
             <event-type-list>
                 <event-type>SimpleEvent</event-type>
             </event-type-list>
             <provider-name>test-rdbms-provider</provider-name>
         </playback-parameters>
         <max-size>10000</max-size>
         <max-threads>2</max-threads>
     </stream>
     ```

   - `Provider Name`: the value of the `rdbms-event-store-provider` child element `name` which corresponds to the `data-source` child element `name`.

     When using the default Berkeley database provider, select `default provider`.

     ```
     <data-source>
         <name>derby1</name>
         <connection-pool-params>
             <initial-capacity>15</initial-capacity>
             <max-capacity>50</max-capacity>
         </connection-pool-params>
         <driver-params>
             <url>jdbc:derby:dbtest1;create=true</url>
             <driver-name>org.apache.derby.jdbc.EmbeddedDriver</driver-name>
         </driver-params>
     </data-source>
     <rdbms-event-store-provider>
         <name>test-rdbms-provider</name>
         <data-source-name>derby1</data-source-name>
     </rdbms-event-store-provider>
     ```

     For more information, see Data Source Configuration.

6. Configure the **Event type list**:

   Use the left and right-pointing arrow buttons to move the events you want played back from the **Available event list** to the **Playback event list**.

   For more information, see Event Type Repository.

7. Decide what you want to do:

   a. To add a new playback entry, click **Add**.

   b. To modify an existing playback, click **Edit**.

c. To delete an existing playback filter, click **Edit**, then click **Clear Filter** under Change Playback Filter Parameters, and click **Save**.

d. To delete an existing playback schedule, click **Edit**, click **Clear Schedule** under Change Playback Schedule Parameters, and click **Save**.

e. To modify an existing playback filter, click **Edit**, change the Filter Start Time and End Time using the calender and clock under Change Playback Filter Parameters section, and click **Save**.

f. To modify an existing playback schedule, click **Edit**, change the Schedule Start Time and End Time using the calender and clock under Change Playback Schedule Parameters section, and click **Save**.

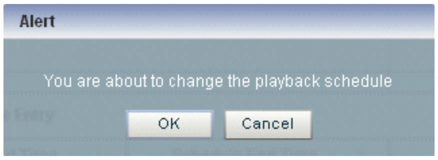
8. Decide what and when you want playback to be:

   a. If you want to create a new playback filter entry:

      - Click the calendar and clock controls in the Change Playback Filter Parameters area to add a start time and end time.
      - Click **Save**.

        An alert dialog appears as Figure 4-25 shows.

        Figure 4-25 Playback Schedule Confirmation Dialog

        You are about to change the playback schedule

      - Click **OK**.

   b. If you want to create a new playback schedule entry:

      - Click the calendar and clock controls in the Change Playback Schedule Parameters area to add a start time and end time and to specify a speed and repeat mode.

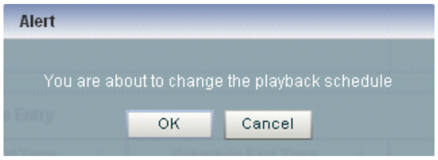
        Speed: the default speed value is 1, which corresponds to normal speed. A value of 2 means that events will be played back 2 times faster than the original record speed. Similarly, a value of 0.5 means that events will be played back at half the speed.

        Repeat: set this to `true` to repeat playback until the scheduled end time or until you stop playback manually; set this to `false` to playback only once.

      - Click **Save**.

        An alert dialog appears as Figure 4-26 shows.

        Figure 4-26 Playback Schedule Confirmation Dialog

        You are about to change the playback schedule

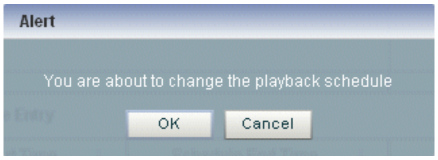
- Click **OK**.

c. If you want to playback events immediately:

- Click **Save**.

  An alert dialog appears as Figure 4-27 shows.

Figure 4-27 Playback Schedule Confirmation Dialog



Alert

You are about to change the playback schedule

- Click **OK**.
- Click **Start**.

Oracle Event Processing Visualizer keeps track of whether a particular stage is currently recording or playing back events; based on this information, the Start and End buttons may be enabled or disabled as appropriate.

When you start playback using Oracle Event Processing Visualizer, Oracle Event Processing uses the event type information from your playback entry and begins playback immediately; the pre-scheduled time, if any, remains unchanged. Use the **End** button to stop playback session immediately.

9. Decide when you want to stop playback:

a. If you scheduled a stop time, event playback will stop at that time.

b. If you want to stop event playback immediately, click **Stop**.

4.4 Trace and Inject Events in the EPN

Using the Event Inspector service and the stream visualizer, you can:

- Trace events on any HTTP pub-sub server channel.
- Trace events on any stage in the EPN on the Event Inspector service dynamic HTTP pub-sub server channel.
- Inject events to any HTTP pub-sub server channel
- Inject events into any stage in the EPN using the Event Inspector service dynamic HTTP pub-sub server channel.

Note:

The Event Inspector service is not for use on a production Oracle Event Processing server. It is for use only during development.

This section describes:

- Trace Events on a Dynamic Channel
- Inject a Simple Event on an Event Inspector Service Dynamic Channel

- Inject a JSON String Event on an Event Inspector Service Dynamic Channel.

For more information, see:

- Event Inspector Service
- HTTP Publish-Subscribe Servers
- Monitor a Channel Stage in the EPN
- *Developing Applications for Oracle Event Processing*.

Alternatively, you can record and playback events as Record and Play Back Events in the EPN describes.

4.4.1 Trace Events on a Dynamic Channel

You can use Oracle Event Processing Visualizer to view the messages that are currently being published to a static channel.

For more information, see HTTP Publish-Subscribe Server Management.

Trace Events on a Dynamic Channel

1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Event Inspector Service.

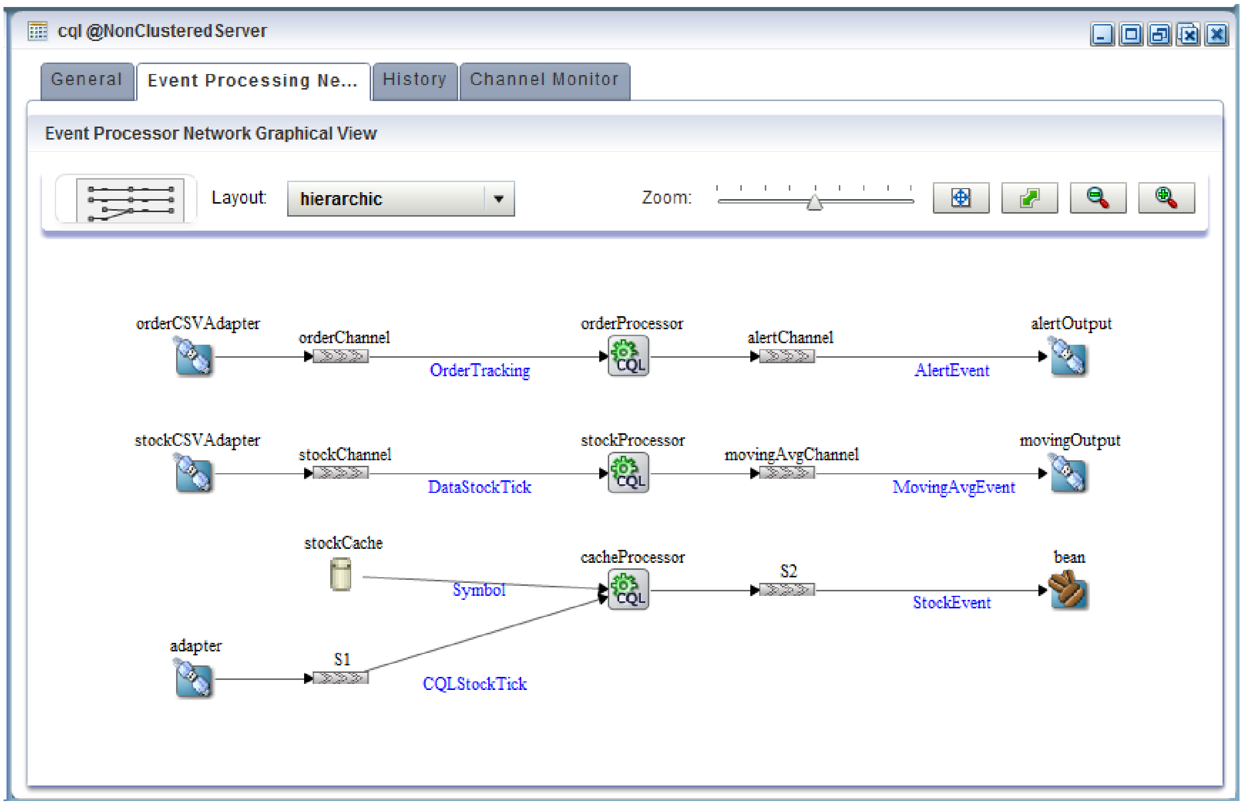
2. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

3. Select ***appname***, where ***appname*** is the name of the application in which you want to trace and inject events.

4. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 4-28 shows.

Figure 4-28 Event Processing Network Panel



5. Select the stage for which you wish to trace events:

   a. To use the EPN diagram:

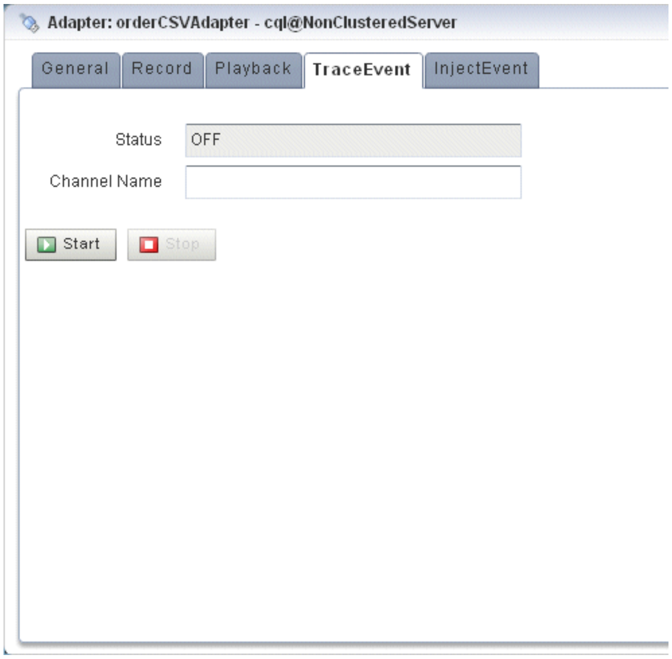
      - Right-click the stage for which you wish to trace events and select **Trace Event**.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace events.
      - Click the stage for which you wish to trace an event.
      - In the right pane, click the **Trace Event** tab

   The Trace Event panel appears as shown in Figure 4-29.

Figure 4-29 The Trace Event Tab: Start



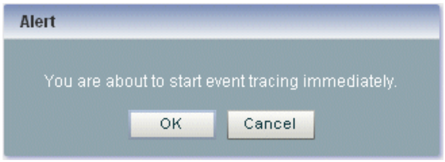
Status: OFF

Channel Name:

6. Click **Start**.

   An alert dialog appears as Figure 4-30 shows.

Figure 4-30 Trace Event Confirmation Dialog



You are about to start event tracing immediately.

7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are written. The Event Inspector service HTTP pub-sub channel is named:

   ```
   /SERVERNAME/APPLICATIONNAME/STAGENAME/output
   ```

   Where:

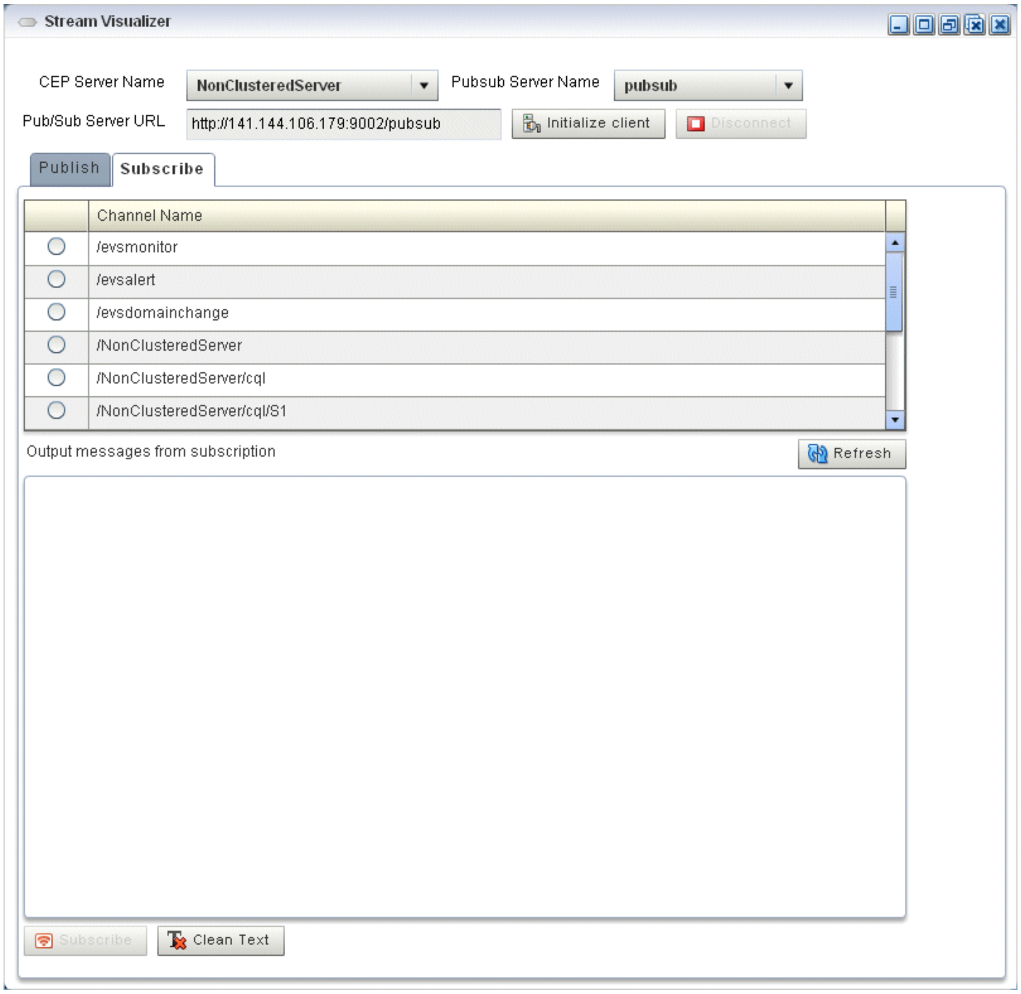
   - *`SERVERNAME`*: the name of the Oracle Event Processing server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle Event Processing application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the Oracle Event Processing application stage you want to trace.

   For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

8. In the top pane, click **ViewStream**.

The Stream Visualizer panel appears as Figure 4-31 shows.

Figure 4-31 Stream Visualizer (ViewStream): Subscribe



CEP Server Name: NonClusteredServer

Pubsub Server Name: pubsub

Pub/Sub Server URL: http://141.144.106.179:9002/pubsub

	Channel Name
○	/evsmonitor
○	/evsalert
○	/evsdomainchange
○	/NonClusteredServer
○	/NonClusteredServer/cql
○	/NonClusteredServer/cql/S1

9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.

10. Click **Initialize Client**.

11. Select the **Subscribe** tab.

12. Click **Refresh**.

    The Subscribe tab is refreshed to show the dynamic channel for your stage.

13. Click the radio button next to the name of the channel to which the Oracle Event Processing server is publishing messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

14. Click **Subscribe**.

    The Output messages received from subscription text box displays events being published to the channel.

Note:

If the Oracle Event Processing Visualizer fails to subscribe to the channel and displays an error in the Debug Messages area such as:

```
14:25:54 GMT-0400: httpFaultHandler(): [RPC Fault faultString="Error #2096"
faultCode="InvokeFailed" faultDetail="null"]
```

Then, confirm the following:

- Have you met the Oracle Event Processing Visualizer Adobe Flash and browser prerequisites? See Prerequisites.

---

15. To clear the Output messages received from subscription text, click **Clear Text**.

16. To unsubscribe from the channel, click **Unsubscribe**.

17. To stop event trace, select the stage you configured to trace events:

    a. To use the EPN diagram:

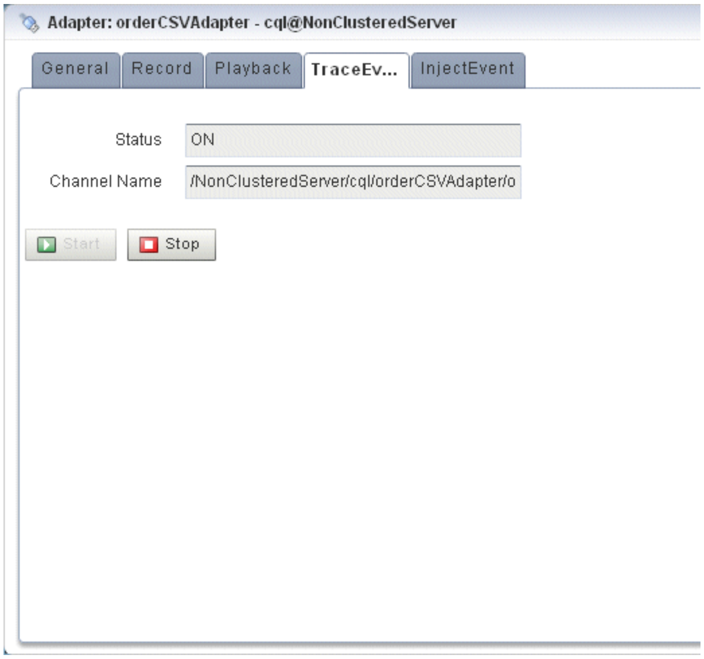
    - Right-click the stage for which you wish to trace an event and select **Trace Event**.

    b. To use the domain tree:

    - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace and inject events.
    - Click the stage for which you wish to trace an event.
    - In the right pane, click the **Trace Event** tab

    The Trace Event panel appears as shown in Figure 4-32.

Figure 4-32 The Trace Event Tab: Stop



Adapter: orderCSVAdapter - cql@NonClusteredServer

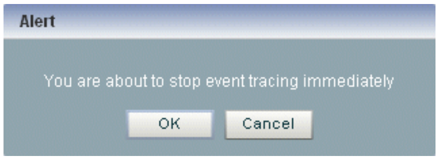
Status: ON

Channel Name: /NonClusteredServer/cql/orderCSVAdapter/o

18. Click **Stop**.

    An alert dialog appears as Figure 4-33 shows.

    Figure 4-33 Trace Event Stop Confirmation Dialog

    Alert

    You are about to stop event tracing immediately

19. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

4.4.2 Inject a Simple Event on an Event Inspector Service Dynamic Channel

You can inject a single, simple event by type into any stage in the EPN using the Event Inspector service dynamic channel.

The Oracle Event Processing Visualizer only supports simple event types whose properties are all simple Java types without nested Java objects. Event properties must be restricted to the following types:

- primitive Java types
- Java array or collection with simple Java type values
- `Date`
- `BigDecimal`
- `BigInteger`

Alternatively, you can inject an event with more complex properties by specifying the event as a JSON message. For more information, see Inject a JSON String Event on an Event Inspector Service Dynamic Channel.

Note:

The Event Inspector service is not for use on a production Oracle Event Processing server. It is for use only during development.

Inject a Simple Event on an Event Inspector Service Dynamic Channel

1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Event Inspector Service.

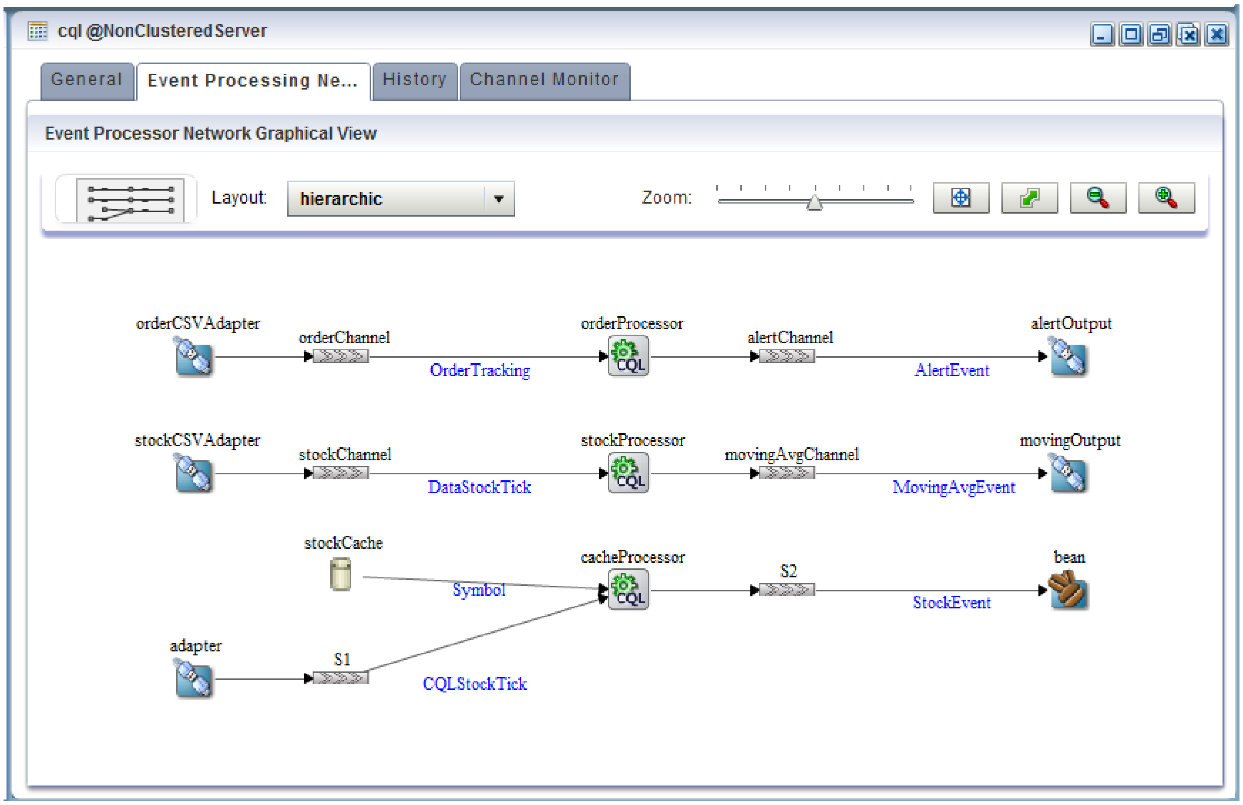
2. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

3. Select ***appname***, where ***appname*** is the name of the application in which you want to trace and inject events.

4. In the right pane, click the **Event Processing Network** tab.

The Event Processor Network panel is displayed as Figure 4-34 shows.

Figure 4-34 Event Processing Network Panel



5. Select the stage for which you wish to inject events:

   a. To use the EPN diagram:

      - Right-click the stage for which you wish to inject an event and select **Inject Event**.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace and inject events.

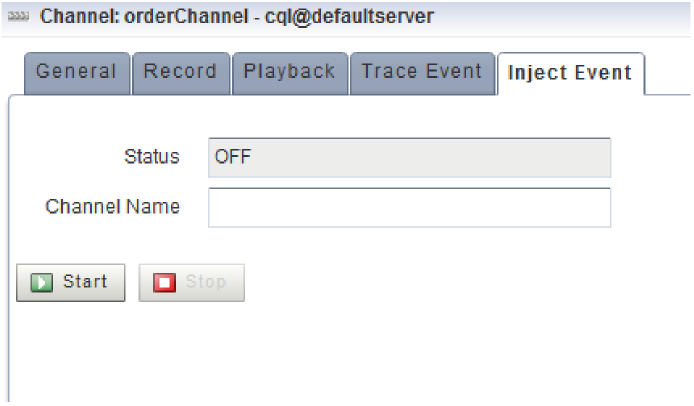
      - Click the stage for which you wish to inject an event.

        Make sure you select a stage that is an event sink.

      - In the right pane, click the **Inject Event** tab

   The Inject Event panel appears as shown in Figure 4-35.

Figure 4-35 The Inject Event Tab: Start



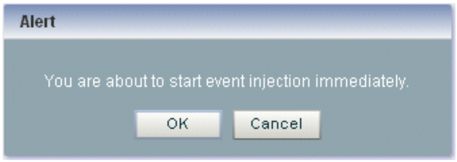
Status: OFF

Channel Name:

6. Click **Start**.

   An alert dialog appears as Figure 4-36 shows.

Figure 4-36 Inject Event Start Confirmation Dialog



You are about to start event injection immediately.

7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are injected. The Event Inspector service HTTP pub-sub channel is named:

   ```
   /SERVERNAME/APPLICATIONNAME/STAGENAME/input
   ```

   Where:

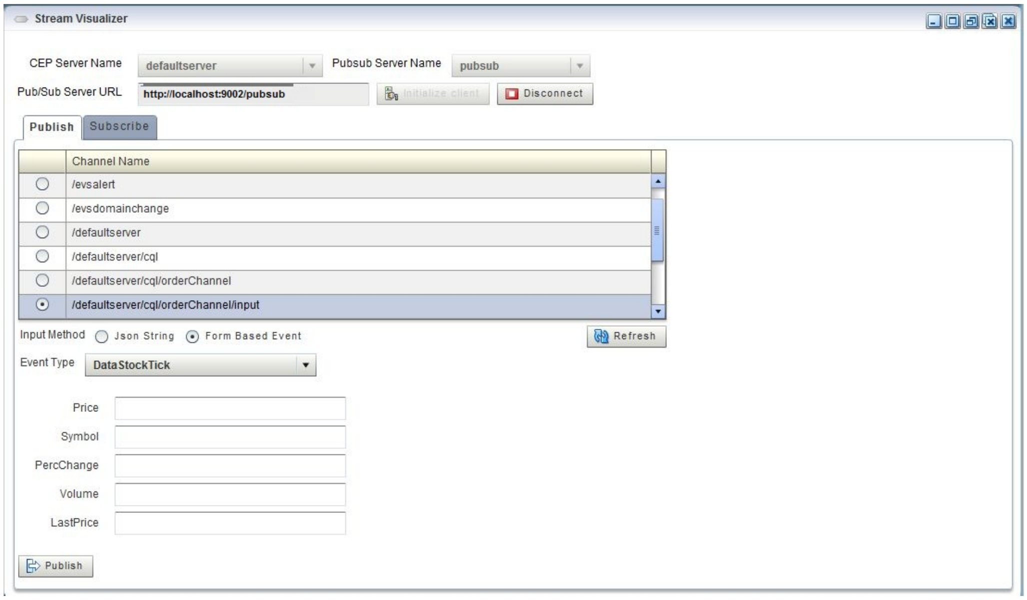
   - *`SERVERNAME`*: the name of the Oracle Event Processing server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle Event Processing application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the Oracle Event Processing application stage you want to trace.

   For example: `/defaultserver/cql/orderChannel/input`.

8. In the top pane, click **ViewStream**.

   The Stream Visualizer panel appears as Figure 4-37 shows.

Figure 4-37 Stream Visualizer (ViewStream): Publishing Simple Events



CEP Server Name: defaultserver

Pubsub Server Name: pubsub

Pub/Sub Server URL: http://localhost:9002/pubsub

	Channel Name
○	/evsalert
○	/evsdomainchange
○	/defaultserver
○	/defaultserver/cql
○	/defaultserver/cql/orderChannel
●	/defaultserver/cql/orderChannel/input

Input Method ○ Json String ● Form Based Event

Event Type: DataStockTick

Price

Symbol

PercChange

Volume

LastPrice

9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.

10. Click **Initialize Client**.

11. Select the **Publish** tab.

12. Click **Refresh**.

    The Publish tab is refreshed to show the dynamic channel for your stage.

13. Click the radio button next to the name of the channel to which the Oracle Event Processing server will inject messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

14. From the Input Method radio button group, select **Form Based Event**.

15. From the **Event Type** pull down menu, select an event type.

    The View Stream panel updates to list the attributes of the event type you select.

    For example, Figure 4-37 shows the attributes for the `DataStockTick` event, which include:

    - `Price`
    - `Symbol`
    - `PercChange`
    - `Volume`
    - `LastPrice`

> **Note:**
>
> The View Stream Event Type pull down menu lists all the events defined in the event type repository that meet the restrictions that Inject a Simple Event on an Event Inspector Service Dynamic Channel describes. To inject a more complex events, see Inject a JSON String Event on an Event Inspector Service Dynamic Channel.

16. Configure the attributes for the event type you selected.

    Hover your mouse over an attribute field to display a tool tip that indicates the data type for the attribute.

17. Click **Publish**.

    Oracle Event Processing Visualizer publishes the event on the Event Inspector service dynamic channel and it is received and processed by the stage that the channel identifies.

18. To stop event injection, select the stage you configured for event injection:

    a. To use the EPN diagram:

    - Right-click the stage for which you wish to inject an event and select **Inject Event**.

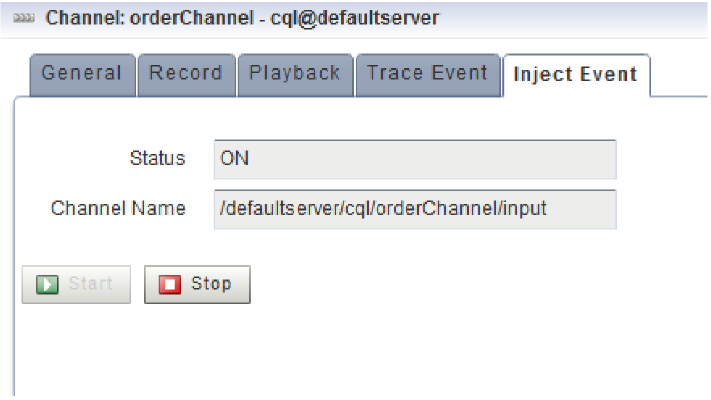
    b. To use the domain tree:

    - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace and inject events.
    - Click the stage for which you wish to inject an event.

      Make sure you select a stage that is an event sink.
    - In the right pane, click the **Inject Event** tab

    The Inject Event panel appears as shown in Figure 4-38.

Figure 4-38 The Inject Event Tab: Stop



Channel: orderChannel - cql@defaultserver

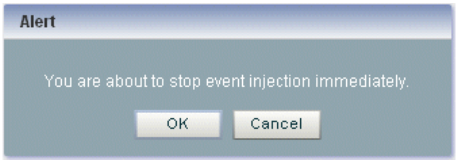
Status: ON

Channel Name: /defaultserver/cql/orderChannel/input

19. Click **Stop**.

    An alert dialog appears as Figure 4-39 shows.

Figure 4-39 Inject Event Stop Confirmation Dialog



Alert

You are about to stop event injection immediately.

20. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

4.4.3 Inject a JSON String Event on an Event Inspector Service Dynamic Channel

You can inject a single event directly to the HTTP pub-sub channel as a JSON-formatted character string. You can use any event property that JSON can represent.

For details on the Event Inspector service JSON event structure and mandatory attributes, see *Developing Applications for Oracle Event Processing*.

Alternatively, you can inject a simple, pre-existing event. For more information, see Inject a Simple Event on an Event Inspector Service Dynamic Channel.

Note:

The Event Inspector service is not for use on a production Oracle Event Processing server. It is for use only during development.

Inject an Event as a JSON String on an Event Inspector Service Dynamic Channel

1. Optionally, configure the HTTP pub-sub server to use to trace events.

   See Event Inspector Service.

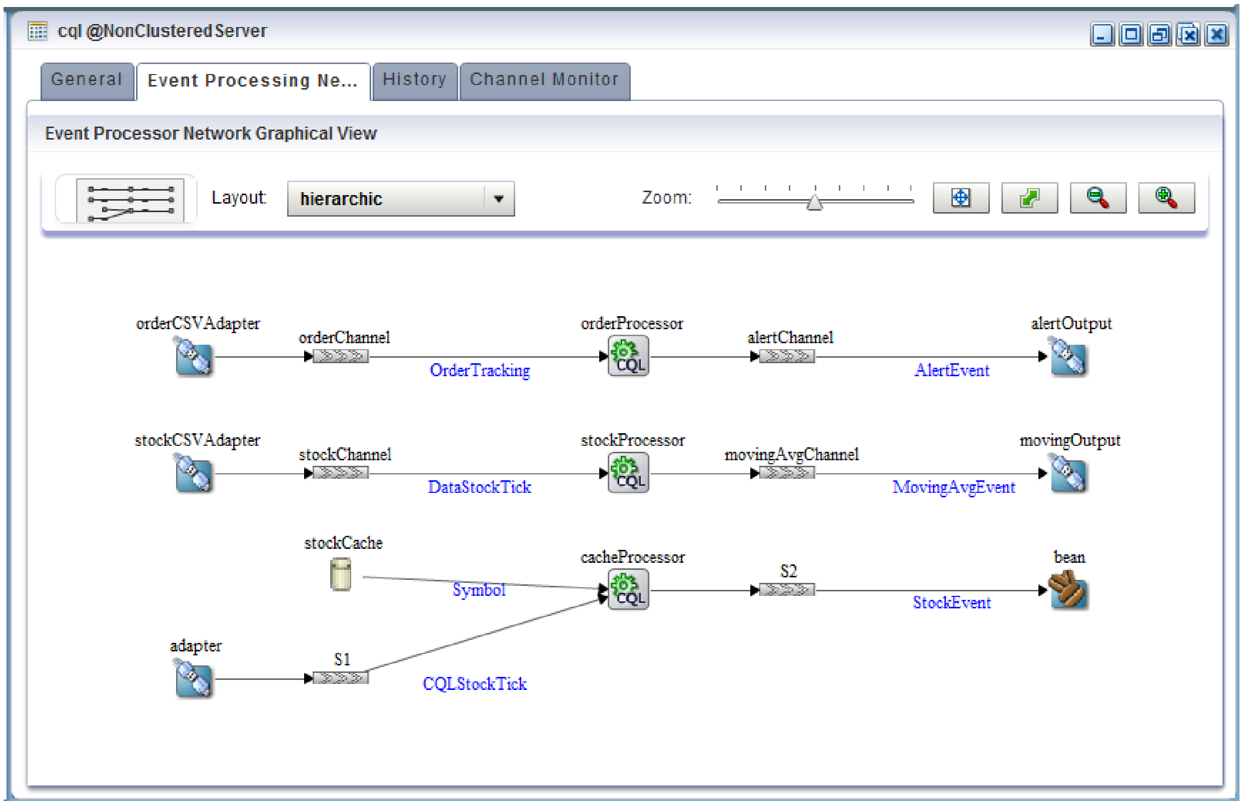
2. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

3. Select ***appname***, where ***appname*** is the name of the application in which you want to trace and inject events.

4. In the right pane, click the **Event Processing Network** tab.

   The Event Processor Network panel is displayed as Figure 4-40 shows.

Figure 4-40 Event Processing Network Panel



5. Select the stage for which you wish to inject events:

   a. To use the EPN diagram:

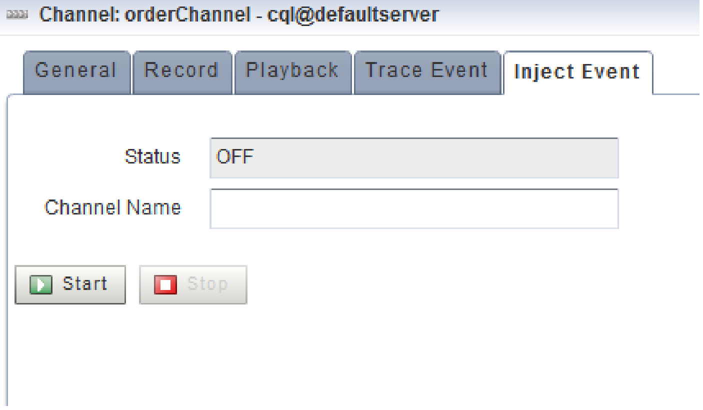
      - Right-click the stage for which you wish to inject an event and select **Inject Event**.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace and inject events.
      - Click the stage for which you wish to inject an event.
      - In the right pane, click the **Inject Event** tab

   The Inject Event panel appears as shown in Figure 4-41.

Figure 4-41 The Inject Event Tab: Start



Channel: orderChannel - cql@defaultserver

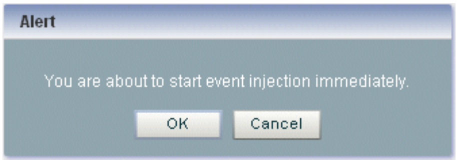
Status: OFF

Channel Name:

6. Click **Start**.

   An alert dialog appears as Figure 4-42 shows.

Figure 4-42 Inject Event Start Confirmation Dialog



Alert

You are about to start event injection immediately.

7. Click **OK**.

   The Status field reads `ON`.

   The Channel Name field shows the dynamic channel on which events passing through this stage are injected. The Event Inspector service HTTP pub-sub channel is named:

```
/SERVERNAME/APPLICATIONNAME/STAGENAME/input
```

   Where:

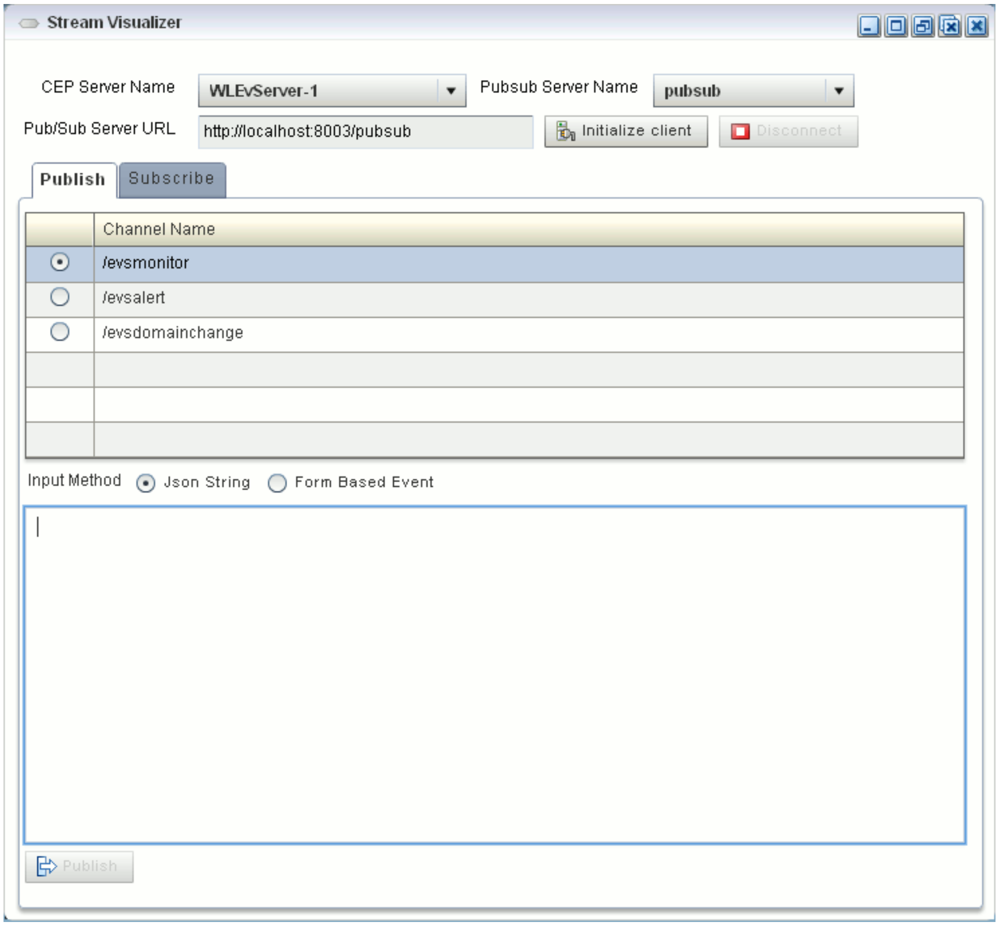
   - *`SERVERNAME`*: the name of the Oracle Event Processing server on which the application and stage you want to trace are executing.
   - *`APPLICATIONNAME`*: the name of the Oracle Event Processing application that owns the stage you want to trace.
   - *`STAGENAME`*: the name of the Oracle Event Processing application stage you want to trace.

   For example: `/NonClusteredServer/cql/orderCSVAdapter/input`.

8. In the top pane, click **ViewStream**.

   The Stream Visualizer panel appears as Figure 4-43 shows.

Figure 4-43 Stream Visualizer (ViewStream): Publishing JSON Events



CEP Server Name: WLEvServer-1

Pubsub Server Name: pubsub

Pub/Sub Server URL: http://localhost:8003/pubsub

	Channel Name
●	/evsmonitor
○	/evsalert
○	/evsdomainchange

Input Method ● Json String ○ Form Based Event

9. In the **PubSub Server Name** pull down menu, select the name of the HTTP pub-sub server to use to trace events.

10. Click **Initialize Client**.

11. Select the **Publish** tab.

12. Click **Refresh**.

    The Publish tab is refreshed to show the dynamic channel for your stage.

13. Click the radio button next to the name of the channel to which the Oracle Event Processing server will inject messages.

    For example: `/NonClusteredServer/cql/orderCSVAdapter/output`.

14. From the Input Method radio button group, select **Json String**.

15. Enter a JSON-formatted string in the text field.

```
{
    "event-type": "myEventType",
    "operation": "insert",
    "binding": "outbound",
    "value":{
        "firstname": "Jane",
        "lastname": "Doe",
        "phone": {
            "code": 12345,
            "number": "office"
        },
    }
}
```

For complete details on the Event Inspector service JSON event structure and mandatory attributes, see *Developing Applications for Oracle Event Processing*.

16. Click **Publish**.

    Oracle Event Processing Visualizer publishes the event on the Event Inspector service dynamic channel and it is received and processed by the stage that the channel identifies.

17. To stop event injection, select the stage you configured for event injection:

    a. To use the EPN diagram:

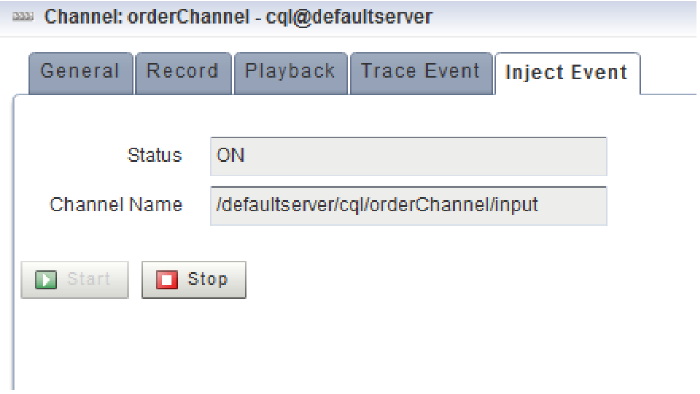
       - Right-click the stage for which you wish to inject an event and select **Inject Event**.

    b. To use the domain tree:

       - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application in which you want to trace and inject events.
       - Click the stage for which you wish to inject an event.
       - In the right pane, click the **Inject Event** tab

    The Inject Event panel appears as shown in Figure 4-44.

    Figure 4-44 The Inject Event Tab: Stop



Channel: orderChannel - cql@defaultserver

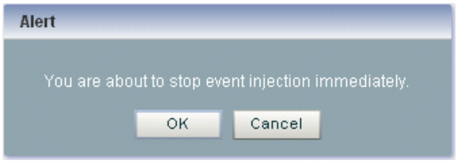
Status: ON

Channel Name: /defaultserver/cql/orderChannel/input

18. Click **Stop**.

    An alert dialog appears as Figure 4-45 shows.

    Figure 4-45 Inject Event Stop Confirmation Dialog



Alert

You are about to stop event injection immediately.

19. Click **OK**.

    The Status field reads `OFF`.

    The Channel Name field is blank.

4.5 Monitor a Channel Stage in the EPN

You can use Oracle Event Processing Visualizer to monitor any channel stage (stream) of the event processing network (EPN) of an application.

Oracle Event Processing defines the `size` metric so that you can monitor the number of events in the channel queue.

Monitor a Channel in the EPN

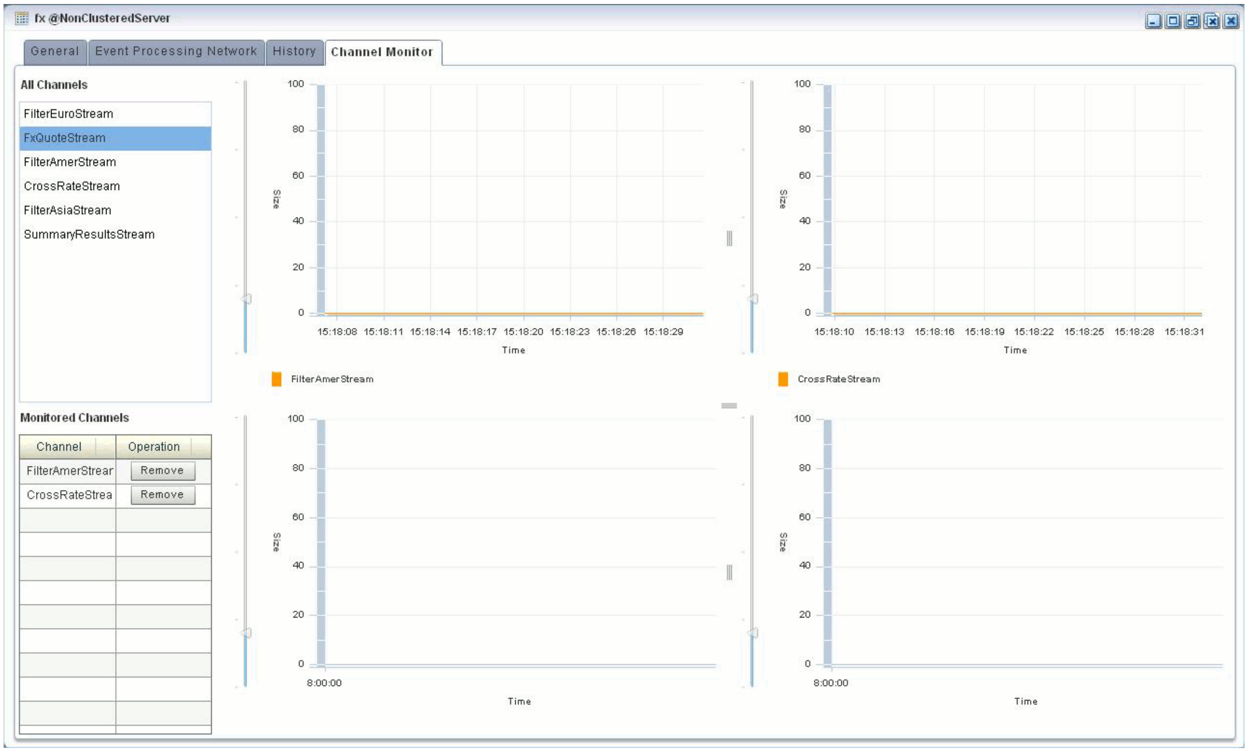
1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application you want to monitor.

3. In the right pane, click the **Channel Monitor** tab.

   The Channel Monitor tab appears as Figure 4-46 shows.

Figure 4-46 Channel Monitor Tab



Channel	Operation
FilterAmerStrear	Remove
CrossRateStrea	Remove

4. To monitor a channel, click on a channel in the **All Channels** list and drag and drop it onto one of the four graphs in the Channel Monitor tab.

   The channel name is added to a color-coded legend below the graph. The color corresponds to the channel's line in the graph.

   The channel name is also added to the Monitored Channels list.

You can display all channels on one graph or distribute channels amongst the four graphs in any combination.

5. To change the scale of the graphs, click on the slider to the left of the vertical axis and drag it up or down.

6. To stop monitoring a channel, click the **Remove** button associated with it in the **Monitored Channels** list.

4.6 Monitor the Throughput and Latency of an EPN Stage or Path

You can use Oracle Event Processing Visualizer to monitor the entry and exit points of a stage, or a specified path, of the event processing network (EPN) of an application. Oracle Event Processing defines the following metrics that you can monitor for each stage or path:

- `Throughput`: The number of events processed by the stage.
- `Average Latency`: The average amount of time it takes an event to pass through a specified path of the EPN.
- `Maximum Latency`: The maximum amount of time it takes an event to pass through a specified path of the EPN.
- `Average Latency Threshold`: Calculates the average latency values greater than the specified threshold value for specified start and end points.

To Monitor the Throughput and Latency of an EPN Stage or Path

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing server instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application for which you want to monitor throughput and latency.

3. Select the stage you wish to view and configure:

   a. To use the EPN diagram:

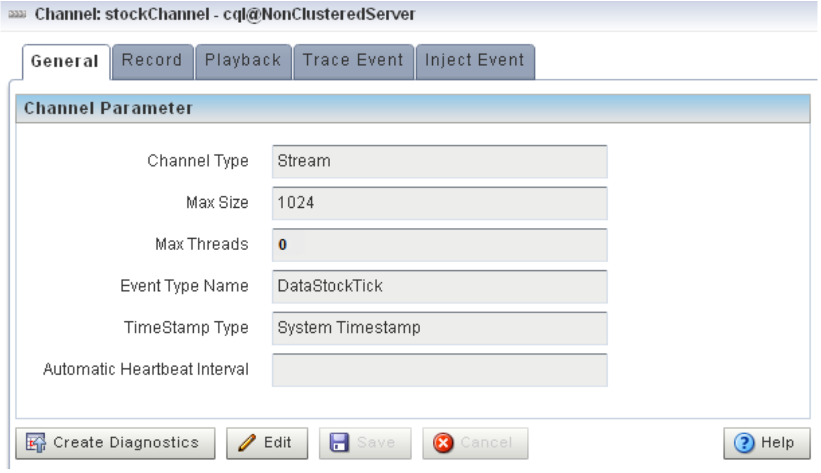
      - Click the **Event Processing Network** tab.
      - Double-click the stage you wish to monitor or the first stage in the path that you want to monitor.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to monitor.
      - Click the stage you wish to monitor or the first stage in the path that you want to monitor.
      - In the right pane, click the **General** tab

   The stage's General tab appears. For example, Figure 4-47 shows the General tab for a channel.

Figure 4-47 General Tab for Channel Stage



Channel Type: Stream
Max Size: 1024
Max Threads: 0
Event Type Name: DataStockTick
TimeStamp Type: System Timestamp
Automatic Heartbeat Interval:

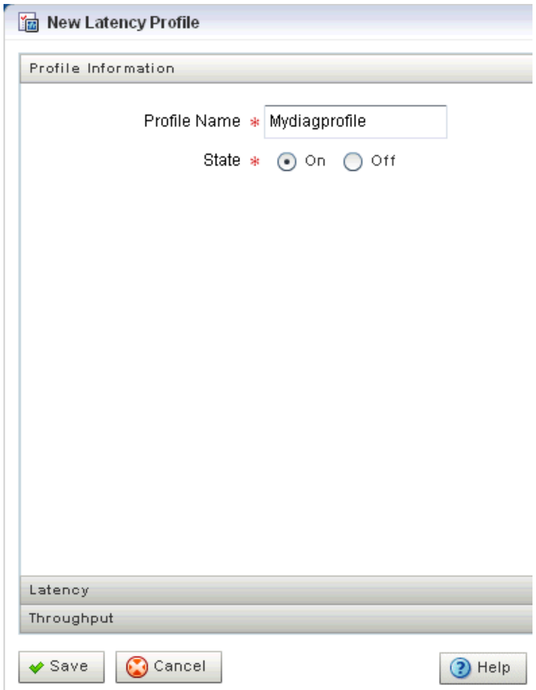
4. Click the **Create Diagnostics** button.

   A New Latency Profile accordion menu appears as shown in Figure 4-48. It contains the following tabs:

   - Profile Information
   - Latency
   - Throughput

5. Click the **Profile Information** tab in the accordion menu as shown in Figure 4-48 and enter the information that Table 4-5 lists.

Figure 4-48 New Latency Profile Screen - Profile Information Tab



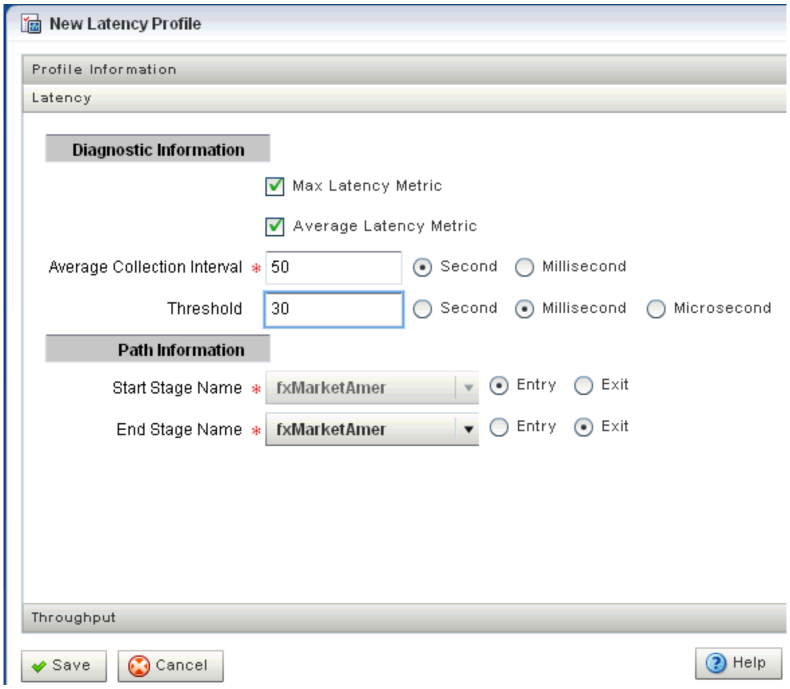
Profile Name *: Mydiagprofile
State *: ● On ○ Off

Table 4-5 Profile Information Attributes

Attribute	Description
`Profile Name`	The name of the diagnostic profile that you want to create.
`State`	Select **On** if you want to enable the profile for immediate use. Select **Off** if you do not want to enable the profile for immediate use.

6. Click the **Latency** tab in the accordion menu as shown in Figure 4-49 and enter the information that Table 4-6 lists.

Figure 4-49 New Latency Profile Screen - Latency Tab



☑ Max Latency Metric

☑ Average Latency Metric

Average Collection Interval * 50 ● Second ○ Millisecond

Threshold 30 ○ Second ● Millisecond ○ Microsecond

Start Stage Name * fxMarketAmer ● Entry ○ Exit

End Stage Name * fxMarketAmer ○ Entry ● Exit

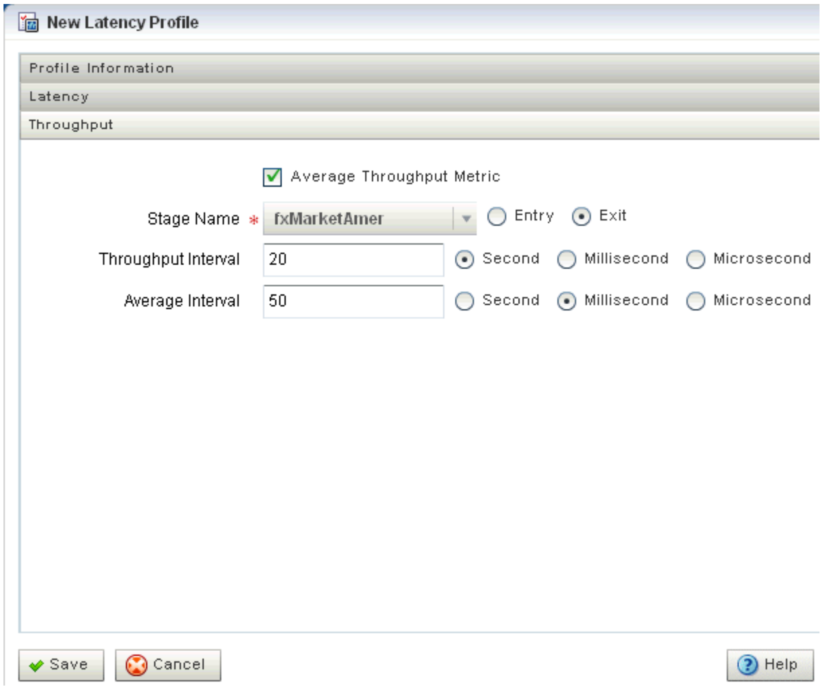
Table 4-6 Latency Attributes

Attribute	Description
`Max Latency Metric`	Specifies that you want to monitor the maximum amount of time it takes for events to flow through a stage or a subset of the event processing network (path).
`Average Latency Metric`	Specifies that you want to monitor the average amount of time it takes for events to flow through a stage or a subset of the event processing network (path)
`Average Collection Interval`	Specifies the time interval for which you want to gather diagnostic data. In other words, the sliding window across which the Average Latency Metric is computed.

Attribute	Description
`Threshold`	Specify a value if you are interested in only receiving latency metrics that exceed a certain value. For example, if you would like to know when the latency exceeds 250 ms, specify a `Threshold` value of 250 ms. This value is enabled only if you specify **Average Latency Metric**.
`Start Stage Name`	Specifies the name that you have provided for the start stage. If you want to monitor just the current stage, rather than a path in the EPN, then set the **Start Stage Name** and **End Stage Name** to the name of the current stage. Select **Entry** for the **Start Stage Name** option and **Exit** for the **End Stage Name**.
`End Stage Name`	Specifies the name that you have provided for the end stage. If you want to monitor a path in the EPN, Oracle Event Processing Visualizer assumes that the current stage is the start of the path, and thus automatically selects it for the **Start Stage Name** field. Specify whether the start of the path should be the **Entry** or **Exit** of the current stage. Then, select the End Stage Name, or the end of the path you want to monitor, and specify whether the end of the path should be the **Entry** or **Exit** of the stage

7. Click the **Throughput** tab in the accordion menu as shown in Figure 4-50 and enter the information that Table 4-7 lists.

Figure 4-50 New Latency Profile Screen - Throughput Tab



☑ Average Throughput Metric

Stage Name * fxMarketAmer ○ Entry ● Exit

Throughput Interval 20 ● Second ○ Millisecond ○ Microsecond

Average Interval 50 ○ Second ● Millisecond ○ Microsecond

Table 4-7 Throughput Attributes

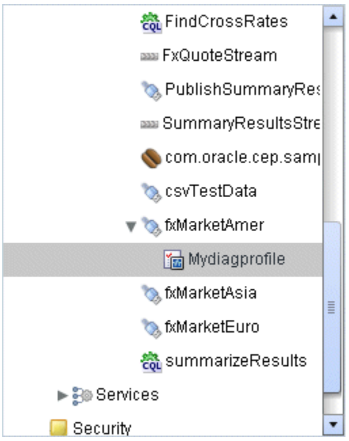
Attribute	Description
`Average Throughput Metric`	Specifies that you want to monitor the average throughput of events flowing through the stage.
`Stage Name`	Specify whether you want to monitor the throughput at the entry or exit of the stage
`Throughput Interval`	Specifies the period of time for which the throughput is calculated. For example, if you specify a `Throughput Interval` of 1 second, the number of events passing through the stage in 1 second will be calculated. This will be averaged over the `Average Interval` you specify. Specify the time unit as **Second**, **Millisecond**, or **Microsecond**.
`Average Interval`	Specifies the interval for gathering the average throughput. For example, if you specify a `Throughput Interval` of 1 second, the number of events passing through the stage in 1 second will be calculated. This will be averaged over the `Average Interval` you specify. Specify the time unit as **Second**, **Millisecond**, or **Microsecond**.

8. Click **Save**.

   A pop-up confirmation dialog appears in the lower-right corner of the panel.

The saved diagnostic profile appears in the left domain tree, under the stage from which you created it, as shown in Figure 4-51.

Figure 4-51 Left Domain Tree - Mydiagprofile



Note:

You can now restart a server or undeploy an application without losing the diagnostic profile you created.

---

9. Click **OK**.

10. Add the diagnostic profile to the Oracle Event Processing Visualizer Dashboard:

    a. Select the diagnostic profile in the left domain tree and click the **Add to Dashboard** button.

    b. Click the **Dashboard** link at the top of Oracle Event Processing Visualizer and drag and drop the diagnostic profile you created from the domain tree in the left pane to the table at the bottom of the right pane.

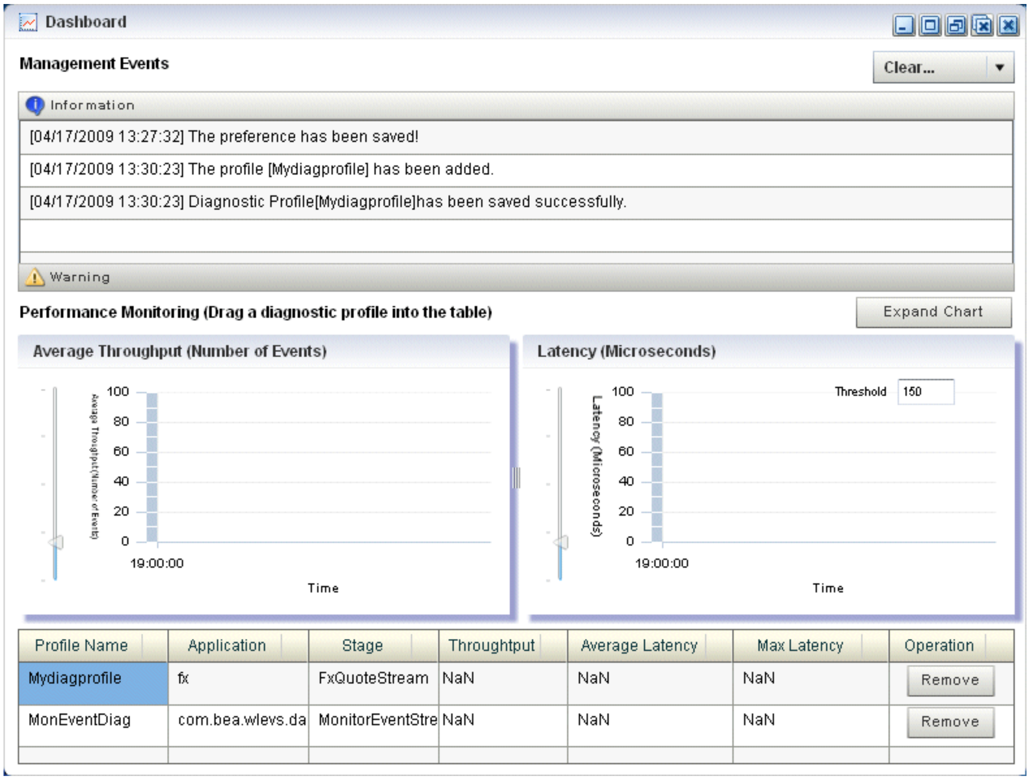
11. Click the **Dashboard** link at the top of Oracle Event Processing Visualizer.

    The Dashboard screen is displayed with the Management Events and Performance Monitoring panes as Figure 4-52 shows.

12. Click the name of the diagnostic profile in the table.

    The latency and throughput information is displayed in the graphs in the middle of the dashboard as Figure 4-52 shows.

Figure 4-52 Oracle Event Processing Visualizer Dashboard With Diagnostic Profiles



[04/17/2009 13:27:32] The preference has been saved!

[04/17/2009 13:30:23] The profile [Mydiagprofile] has been added.

[04/17/2009 13:30:23] Diagnostic Profile[Mydiagprofile]has been saved successfully.

Profile Name	Application	Stage	Throughtput	Average Latency	Max Latency	Operation
Mydiagprofile	fx	FxQuoteStream	NaN	NaN	NaN	Remove
MonEventDiag	com.bea.wlevs.da	MonitorEventStre	NaN	NaN	NaN	Remove

13. To expand the performance monitoring graphs to fill the screen, click **Expand Chart**.

14. To restore the performance monitoring graphs to their original size, click **Restore Chart**.

The Management Events section at the top of the Dashboard displays alerts about the incoming monitoring events.

5

Application Lifecycle

This chapter describes how you can use Oracle Event Processing Visualizer to deploy, suspend, update, and uninstall Oracle Event Processing applications.

This chapter includes the following sections:

- Deploy an Application
- Suspend or Resume an Application
- Update an Application
- Uninstall an Application

For more information, see:

- Application Lifecycle Management
- Application Deployments.

5.1 Deploy an Application

Deploying an application refers to uploading to the server the JAR file that contains the application and then installing it, which makes it available to clients.

Oracle Event Processing internally deploys an application as a two step processes. The first step involves starting the application bundle inside the OSGi container. The second step involves starting and initializing the application. Because the second step is asynchronous in nature, Oracle Event Processing Visualizer does not wait for the completion of the second operation. As soon as the application has successfully started, the Oracle Event Processing Visualizer navigation tree is updated automatically with the new deployment and a successful message will be sent to the dashboard. However, if the application fails to start, you must check for errors on the server side because no messages/updates will occur on the Oracle Event Processing Visualizer.

When you deploy an application, the Oracle Event Processing server starts a new configuration history for it. See Configuration History Management.

Note:

You may only deploy to a group if the target Oracle Event Processing server is part of a multiserver domain (that is, if clustering is enabled). You may not deploy to a group if the target Oracle Event Processing server is part of a standalone-server domain (that is, if clustering is disabled).

This section describes:

- Deploy an Application in a Standalone-Server Domain
- Deploy an Application in a Multiserver Domain.

5.1.1 Deploy an Application in a Standalone-Server Domain

This section describes how to deploy an application to a standalone-server domain. For information on deploying an application in a multiserver domain, see Deploy an Application in a Multiserver Domain.

> **Note:**
>
> Uploading an application JAR file via Oracle Event Processing Visualizer running in the Firefox browser over HTTPS is not supported. For uploading over HTTPS, use Internet Explorer.

Deploy an Application in a Standalone Server Domain

1. In the left pane, navigate to and select the **Deployment** node.

   The Deployed Applications table is displayed in the Deployment screen on the right pane.

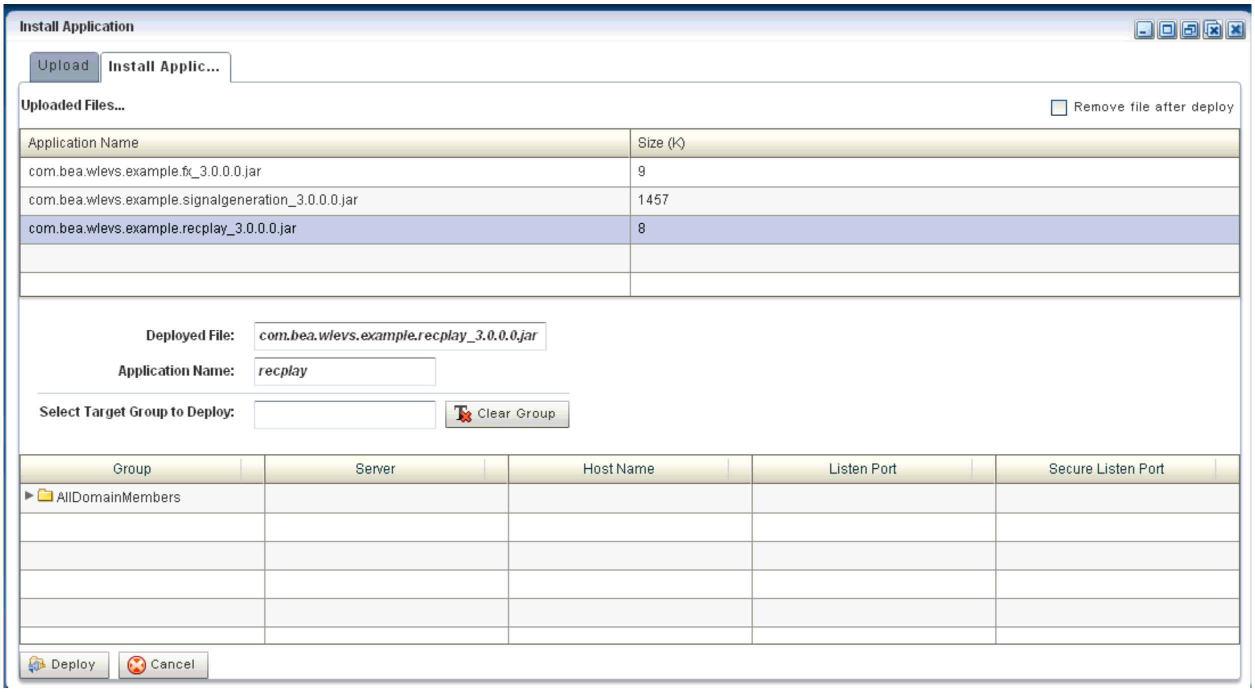
2. Click the **Install** button.

   The Install Application screen is displayed.

3. Click the **Upload** tab.
4. Click the **...** button to open a file browsing window.
5. Browse to the local directory that contains the JAR file of the application you want to install and click **Open**.
6. Click **Upload**.

   In the Install Application tab, the JAR file appears in the table of uploaded files, as shown in Figure 5-1.

Figure 5-1 The Install Application Screen: Standalone-Server Domain



Uploaded Files...

☐ Remove file after deploy

Application Name	Size (K)
com.bea.wlevs.example.fx_3.0.0.0.jar	9
com.bea.wlevs.example.signalgeneration_3.0.0.0.jar	1457
com.bea.wlevs.example.recplay_3.0.0.0.jar	8

Deployed File: *com.bea.wlevs.example.recplay_3.0.0.0.jar*

Application Name: *recplay*

Select Target Group to Deploy:

Group	Server	Host Name	Listen Port	Secure Listen Port
▶ AllDomainMembers				

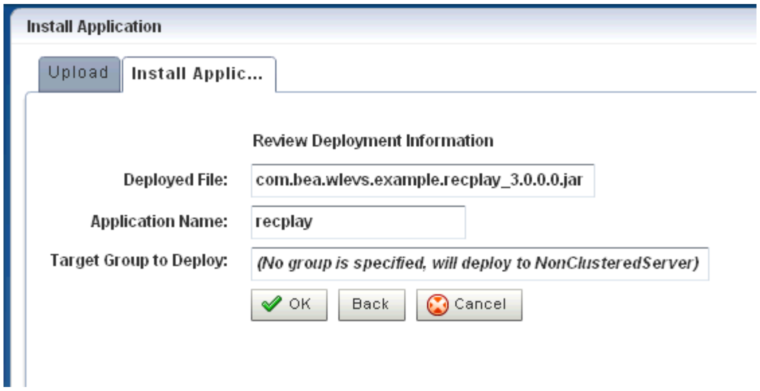
7. If you want the application JAR file removed from the Oracle Event Processing server after deploying, check the **Remove file after deploy** check box.

   In this case, you must upload the application JAR file again if you want to redeploy this application.

8. Select your application in the table and click **Deploy**.

   The Review Deployment Information screen is displayed, as shown in Figure 5-2.

   Figure 5-2 The Review Deployment Information Screen: Standalone-Server Domain



Review Deployment Information

Deployed File: com.bea.wlevs.example.recplay_3.0.0.0.jar

Application Name: recplay

Target Group to Deploy: *(No group is specified, will deploy to NonClusteredServer)*

9. Click **OK**.

   Your application is deployed and ready for use.

5.1.2 Deploy an Application in a Multiserver Domain

This section describes how to deploy an application to a multiserver domain. For information on deploying an application in a standalone-server domain, see Deploy an Application in a Standalone-Server Domain.

For more information, see Multiserver Domains.

> **Note:**
>
> Uploading an application JAR file via Oracle Event Processing Visualizer running in the Firefox browser over HTTPS is not supported. For uploading over HTTPS, use Internet Explorer.

Deploy an Application in a Multiserver Domain

1. In the left pane, navigate to and select the **Deployment** node.

   The Deployed Applications table is displayed in the Deployment screen on the right pane.

2. Click the **Install** button.

   The Install Application screen is displayed.

3. Click the **Upload** tab.

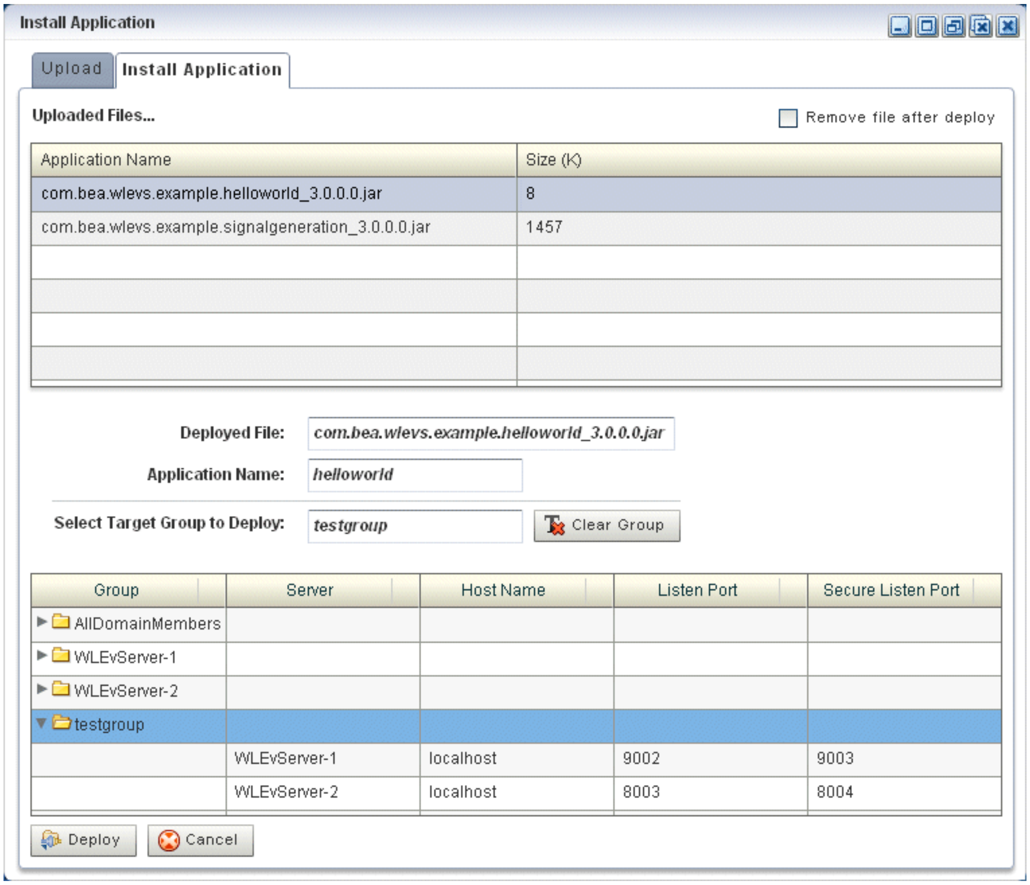
4. Click the **...** button to open a file browsing window.

5. Browse to the local directory that contains the JAR file of the application you want to install and click **Open**.

6. Click **Upload**.

   In the Install Application tab, the JAR file appears in the table of uploaded files, as shown in Figure 5-3.

Figure 5-3 The Install Application Screen: Multiserver Domain



Install Application

Uploaded Files...

□ Remove file after deploy

Application Name	Size (K)
com.bea.wlevs.example.helloworld_3.0.0.0.jar	8
com.bea.wlevs.example.signalgeneration_3.0.0.0.jar	1457

Deployed File: com.bea.wlevs.example.helloworld_3.0.0.0.jar

Application Name: helloworld

Select Target Group to Deploy: testgroup

Group	Server	Host Name	Listen Port	Secure Listen Port
▶ AllDomainMembers				
▶ WLEvServer-1				
▶ WLEvServer-2				
▼ testgroup				
	WLEvServer-1	localhost	9002	9003
	WLEvServer-2	localhost	8003	8004

7. Configure the **Select Target Group to Deploy** field.

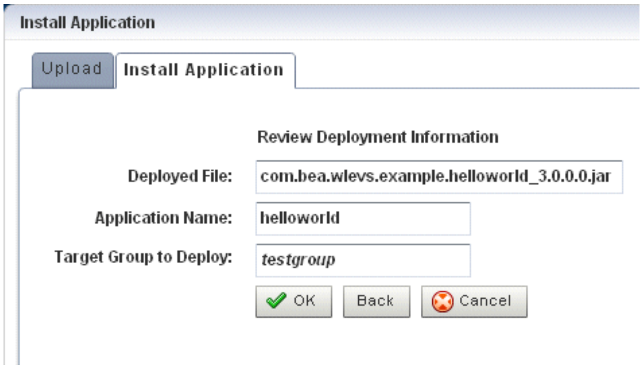
   The group name appears in the **Target Group to Deploy** field.

   To unselect this selection, click the **Clear Group** button.

8. Select your application in the table and click **Deploy**.

   The Review Deployment Information screen is displayed, as shown in Figure 5-4.

Figure 5-4 The Review Deployment Information Screen: Multiserver Domain



Install Application

Review Deployment Information

Deployed File: com.bea.wlevs.example.helloworld_3.0.0.0.jar

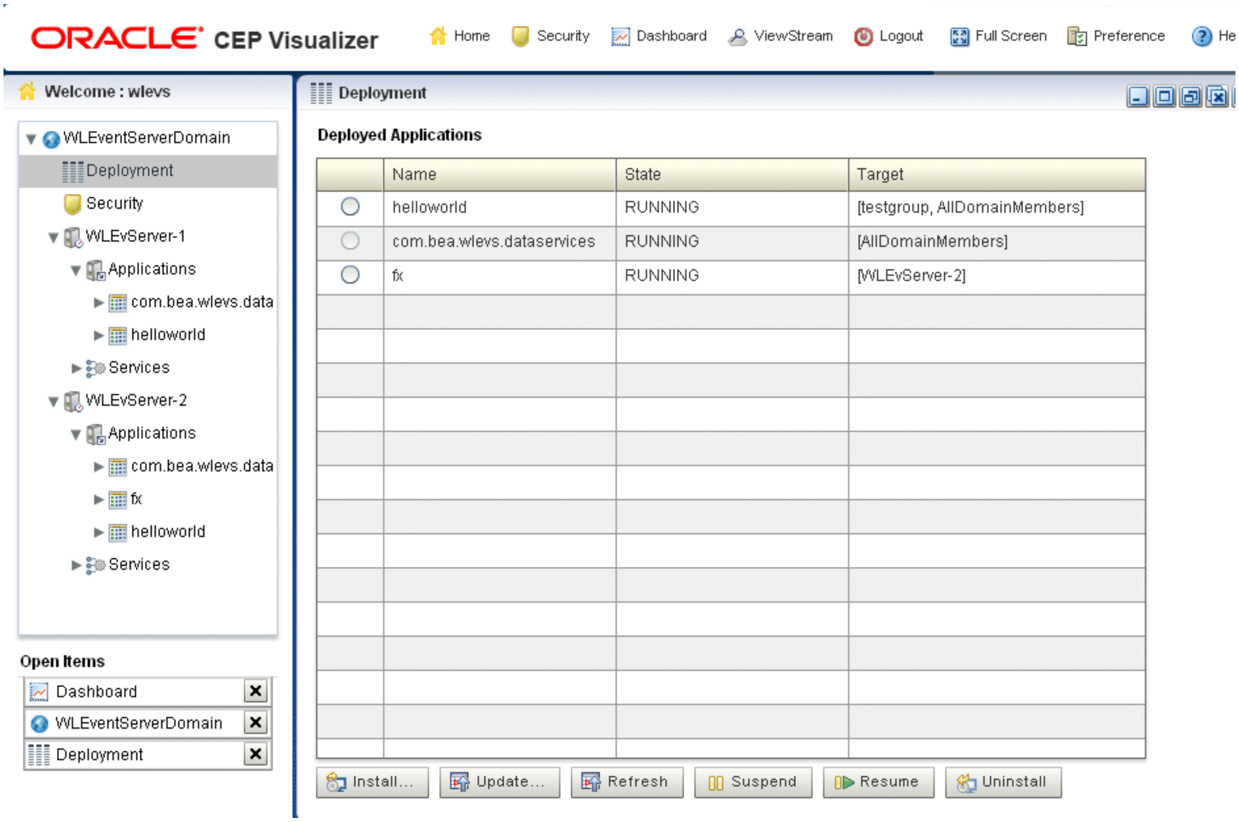
Application Name: helloworld

Target Group to Deploy: testgroup

9. Click **OK**.

Oracle Event Processing Visualizer deploys your application to all servers in the selected target group as Figure 5-5 shows.

Figure 5-5 HelloWorld Application Deployed to All the Servers in Group testgroup



Deployed Applications

	Name	State	Target
○	helloworld	RUNNING	[testgroup, AllDomainMembers]
○	com.bea.wlevs.dataservices	RUNNING	[AllDomainMembers]
○	fx	RUNNING	[WLEvServer-2]

5.2 Suspend or Resume an Application

You can temporarily suspend an application from running and resume the operation using the Suspend and Resume operations in the Deployment screen.

Note:

Suspend and resume apply to installed applications only.

Suspend or Resume an Application

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.

2. Select the application you want to suspend or resume by checking the box to the left of its name.

3. Click **Suspend** to suspend the application or **Resume** to resume a suspended application.

5.3 Update an Application

You can update your currently existing application with your latest application using the Update operation in the Deployment screen.

When you update an application, the Oracle Event Processing server erases the application's existing configuration history and starts a new configuration history for it. See Configuration History Management.

Note:

Uploading an application JAR file via Oracle Event Processing Visualizer running in the Firefox browser over HTTPS is not supported. For uploading over HTTPS, use Internet Explorer.

Update an Application

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.
2. Select the application you want to update by checking the box to the left of its name.
3. Click **Update**. The Update application screen is displayed.
4. Click the **Upload** tab.
5. Click on the "..." button to invoke a file browsing window, browse to the directory that contains the JAR file of application, and click **Open**.
6. Click **Upload**. In the Update Application tab, the JAR file appears in the table of uploaded files.
7. Select your application in the table and click **OK**.

   The existing application is updated with the latest application.

5.4 Uninstall an Application

When you uninstall an application, you completely remove it from the server so that you can no longer access it.

When you uninstall an application, the Oracle Event Processing server erases the application's existing configuration history. See Configuration History Management.

Uninstall an Application

1. In the left pane, navigate to **Deployment** node. The Deployed Applications table is displayed in the Deployment screen on the right pane.
2. Select the application you want to uninstall by checking the box to the left of its name.
3. Click **Uninstall**.

   You have undeployed the application successfully.

6

Oracle CQL Rules

This chapter describes how you can use Oracle Event Processing Visualizer to develop and manage Oracle CQL rules, which you can use to select events as they pass through an event processing network (EPN).

This chapter includes the following sections:

- Query Wizard
- Use the Query Plan.

6.1 Query Wizard

Using the Oracle Event Processing Visualizer, you can manage the Oracle CQL rules in the Oracle CQL processors of an Oracle Event Processing application.

This section describes the following topics:

- Create a Rule with the Query Wizard
- Edit a Rule with the Query Wizard
- Edit a Parameterized Query
- Delete a Rule in an Oracle CQL Processor
- Replace a Rule
- View a Rule
- Configure Oracle CQL Constructs with Query Wizard.

6.1.1 Create a Rule with the Query Wizard

Oracle Event Processing Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL view and query construction. You can drag and drop Oracle CQL elements to construct a view or query or you can drag and drop a complete Oracle CQL view or query template and customize it to suit your needs. The Oracle CQL Query Wizard is designed to allow business users to efficiently create syntactically correct Oracle CQL views and queries and apply them immediately to a deployed Oracle Event Processing application.

Using the Query Wizard, you can create Oracle CQL views and query. You cannot create Oracle CQL streams and relations. Your deployed Oracle Event Processing application must provide the streams and relations. For more information, see *Developing Applications for Oracle Event Processing*.

Note:

An Oracle CQL query will immediately begin outputting events if its input channels provide input events. If you plan to use a query selector on a channel with an upstream Oracle CQL processor, you may observe unwanted query results on the downstream channel between the time you add the query to the upstream Oracle CQL processor and the time you configure the query selector on the downstream channel. For more information, see Channel Properties: Outbound Channel With Query Selector.

Alternatively, you can modify an existing Oracle CQL view or query; see Edit a Rule with the Query Wizard.

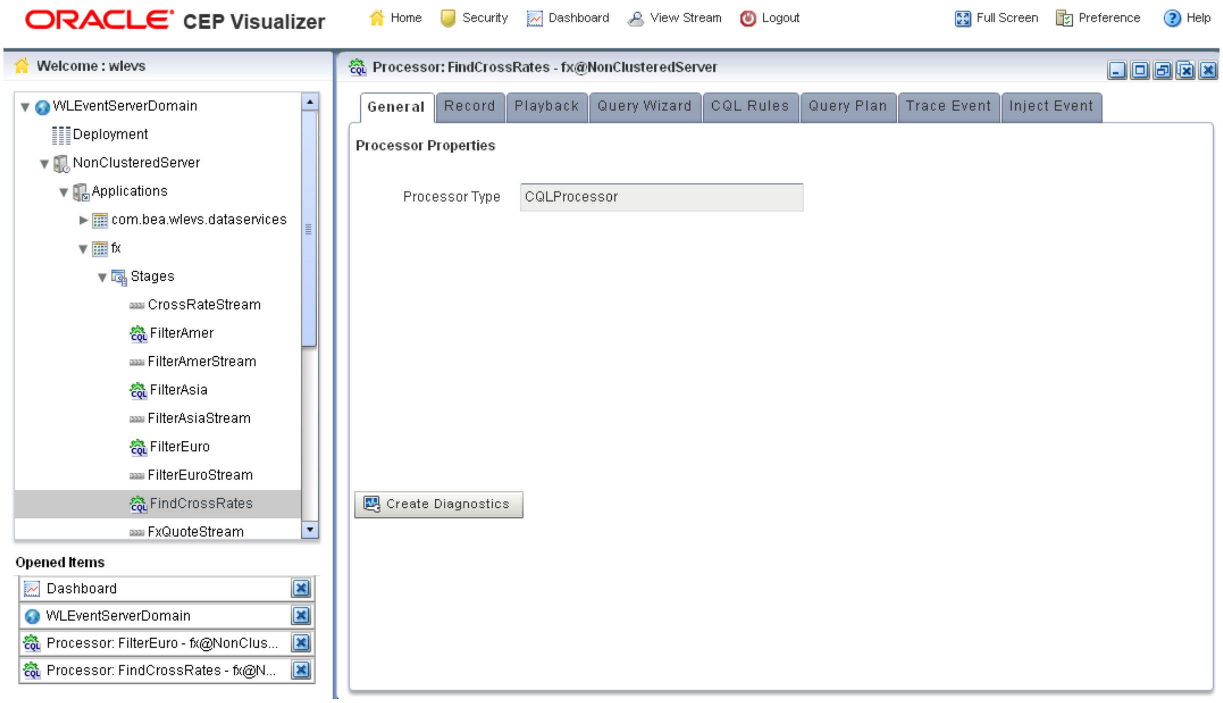
Create a Rule with the Query Wizard

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-1 shows.

Figure 6-1 Oracle CQL Processor Panel

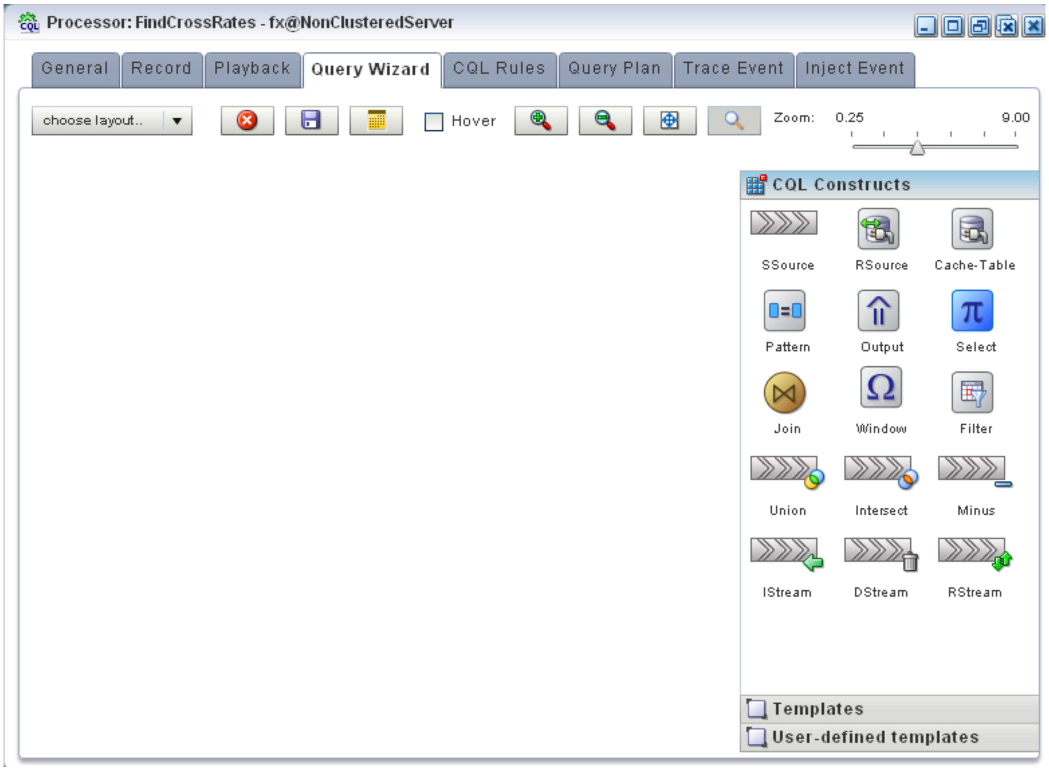


Processor Type: CQLProcessor

4. Click the **Query Wizard** tab.

   The Query Wizard tab appears as Figure 6-2 shows.

Figure 6-2 Oracle CQL Query Wizard



For information on the various tools along the top of the Query Wizard canvas, see Manage the Query Wizard Diagram.

5. Decide how you want to create the Oracle CQL statement:

   a. To construct the query based on an existing Oracle Event Processing server template:

      - Click the **Templates** tab.
      - Select a template and drag and drop it into the canvas.

   b. To construct the query based on an existing user-defined template:

      - Click the **User defined templates** tab.
      - Select a template and drag and drop it into the canvas.

   c. To construct the query one operator at a time or to add additional operators to a template-based Oracle CQL statement:

      - Click the **CQL Constructs** tab.
      - Select a construct and drag and drop it into the canvas.
      - Repeat until you have added the required constructs to the canvas.

6. Configure the connections between constructs:

   a. To connect constructs, click on a source construct and drag to a destination construct.

      Oracle Event Processing Visualizer only allows interconnections that conform to Oracle CQL syntax.

      Figure 6-3 shows a valid interconnection.

      Figure 6-3 Valid Connection

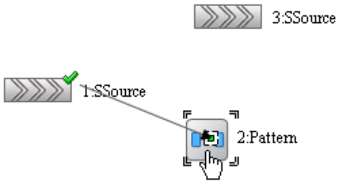
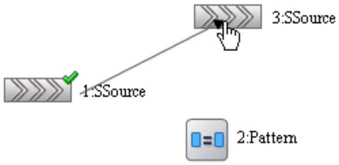


      Figure 6-4 shows an invalid interconnection.

      Figure 6-4 Invalid Connection



      For more information about valid Oracle CQL syntax, see *Developing Applications for Oracle Event Processing*.

   b. To abandon a connection attempt, click the **Esc** key on your keyboard.

   c. To delete an existing connection, click on the connection to select it and then click the **Delete** key on your keyboard.

7. Configure each construct by double-clicking the construct and using the editor dialog that appears.

   A construct icon without a green check mark is not configured.

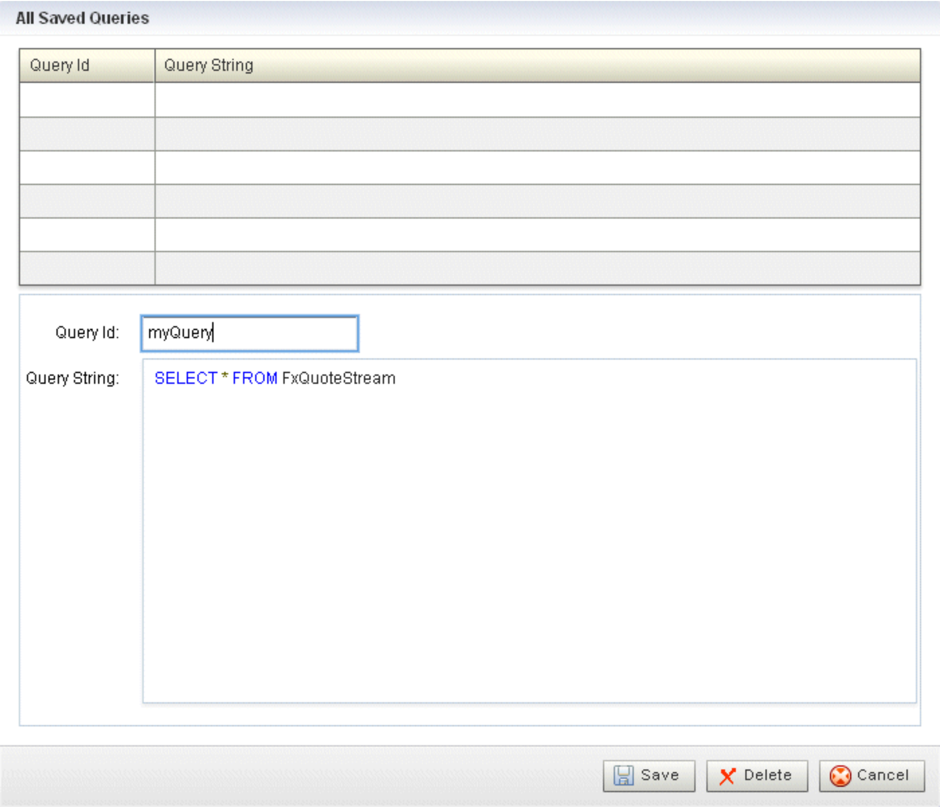
   A construct icon with a green check mark is configured.

8. To save your Oracle CQL statement at any time, click the **Save Query** button.

   The All Saved Queries dialog appears as Figure 6-5 shows.

   The All Saved Queries dialog lists any saved queries for the processor you are configuring.

Figure 6-5 All Saved Queries Dialog: Saving



All Saved Queries

Query Id	Query String

Query Id: myQuery

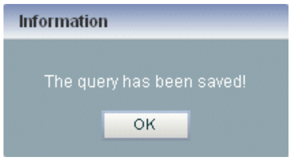
Query String:

```
SELECT * FROM FxQuoteStream
```

9. Enter a query name in the **Query Id** field and click **Save**.

   A confirmation dialog appears as Figure 6-6 shows.

Figure 6-6 Query Save Dialog



Information

The query has been saved!

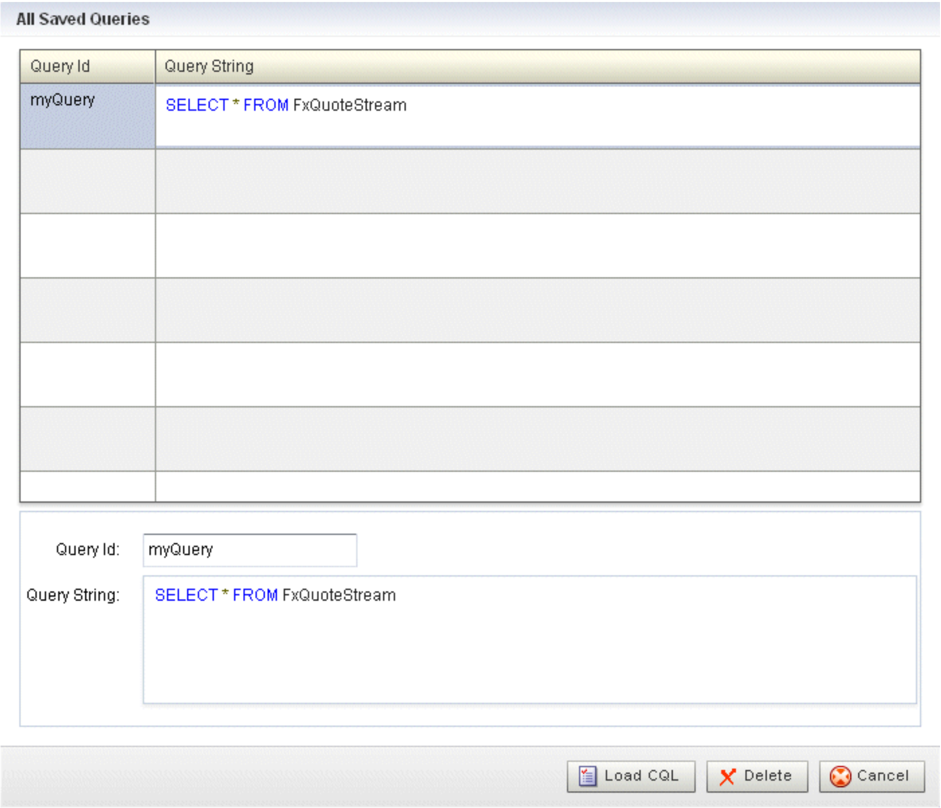
10. Click **OK**.

Your query is saved to the local disk of the computer you are currently using. To reload your query, you must access the Oracle Event Processing Visualizer from the same local host.

11. To reload your Oracle CQL statement after saving, click the **Load Query** button.

    The All Saved Queries dialog appears as Figure 6-7 shows.

Figure 6-7 All Saved Queries Dialog: Loading



All Saved Queries

Query Id	Query String
myQuery	SELECT * FROM FxQuoteStream

Query Id: myQuery

Query String: SELECT * FROM FxQuoteStream

12. Select your query and click **Load CQL**.

    The Oracle CQL statement is loaded to the canvas.

13. To inject your Oracle CQL statement into the Oracle CQL processor, edit the **Output** CQL construct and click **Inject Rule**.

    The rule now appears on the **CQL Rules** tab.

    For more information, see Configure an Output CQL Construct.

14. After you create a query, update the selector for the outbound channel (the channel down-stream from the Oracle CQL processor you created your query on). This will only work with pre-existing event types.

    For more information, see Channel Properties: Outbound Channel .

6.1.2 Edit a Rule with the Query Wizard

Oracle Event Processing Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL query construction. You can select an existing query on the CQL Query tab and edit the query in the Query Wizard. The Oracle CQL Query Wizard is designed to allow business users to efficiently create syntactically correct Oracle CQL queries and apply them to a deployed Oracle Event Processing application.

Query Wizard will import a given rule and display it graphically as a set of Oracle CQL operators. You can go into an individual operator and make changes (for example, to change the select expression) without affecting other parts of the Oracle CQL statement.

You can edit existing Oracle CQL queries and views. You cannot edit Oracle CQL streams and relations. Your deployed Oracle Event Processing application must provide the streams and relations. For more information, see *Developing Applications for Oracle Event Processing*.

Edit a Rule with the Query Wizard:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application you want to use.

3. Select the Oracle CQL processor you wish to use:

   a. To use the EPN diagram:

      - Click the **Event Processing Network** tab.

      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:

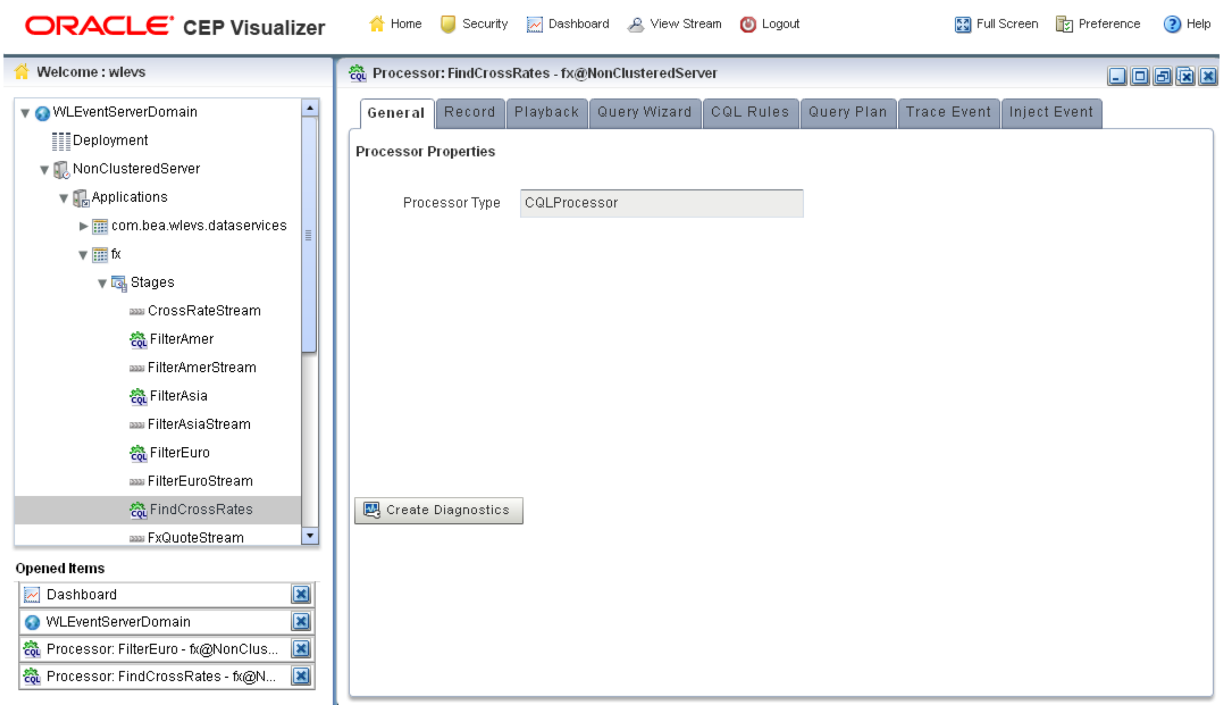
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.

      - Click the Oracle CQL processor you wish to use.

      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-8 shows.

Figure 6-8 Oracle CQL Processor Panel



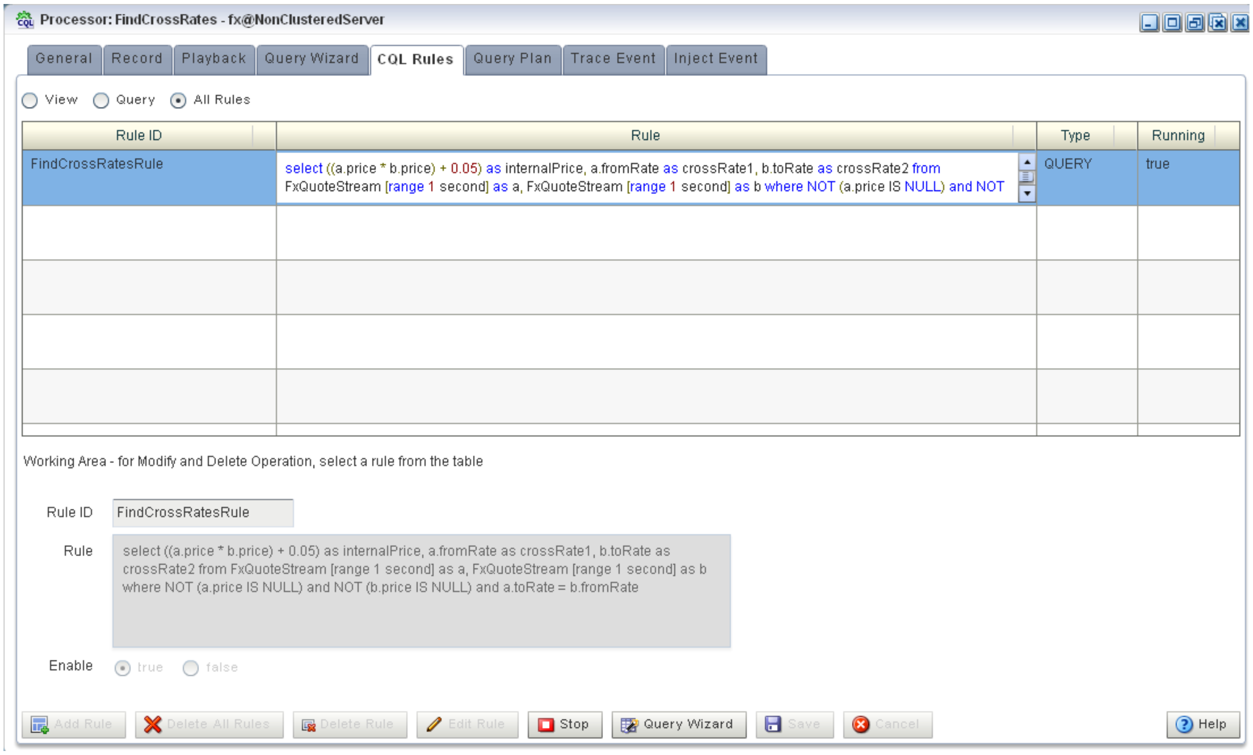
Processor Properties

Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-9 shows.

Figure 6-9 CQL Rules Tab



○ View ○ Query ● All Rules

Rule ID	Rule	Type	Running
FindCrossRatesRule	select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Rule ID: FindCrossRatesRule

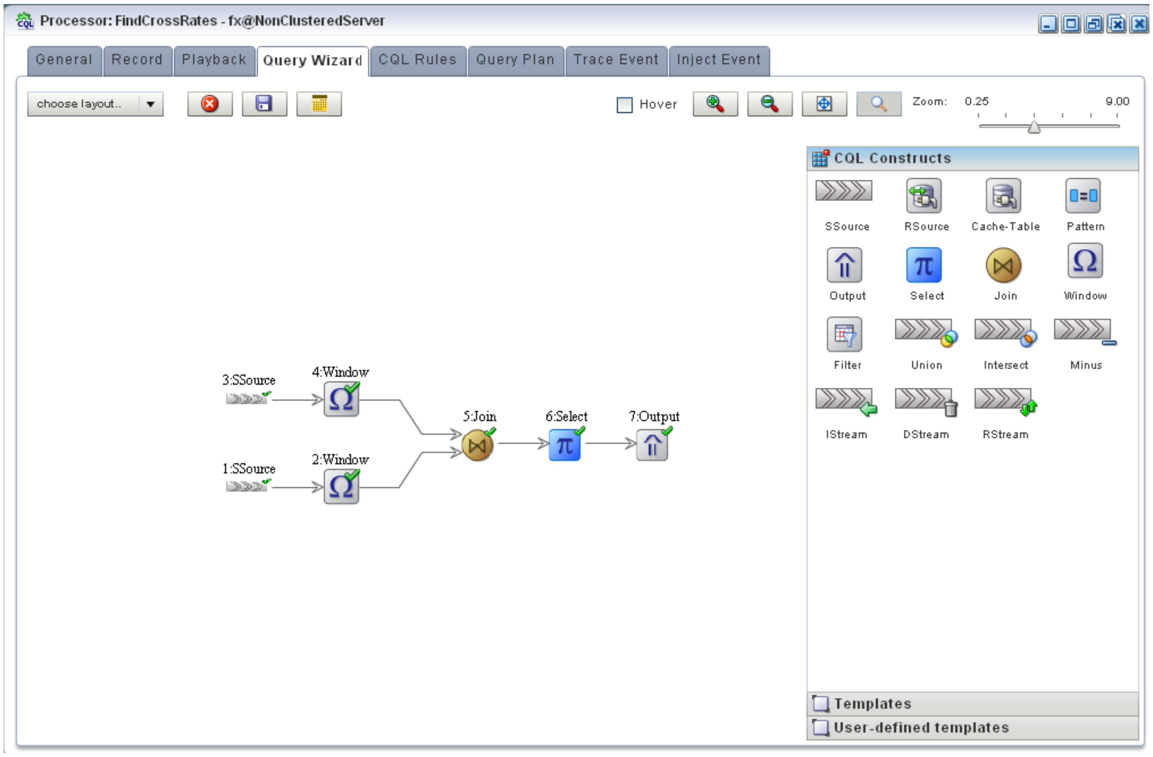
Rule: select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT (b.price IS NULL) and a.toRate = b.fromRate

Enable: ● true ○ false

5. Select the rule you want to edit and click the **Query Wizard** button.

The Query Wizard opens with the selected rule on the canvas as Figure 6-10 shows.

Figure 6-10 Query Wizard: With Existing Rule



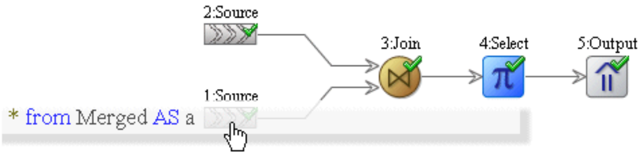
For information on the various tools along the top of the Query Wizard canvas, see Manage the Query Wizard Diagram.

6. To view the Oracle CQL statements associated with any operator:

   - Ensure that the **Hover** check box is checked.
   - Hover the mouse pointer over an operator.

   The Oracle CQL statement appears as Figure 6-11 shows.

Figure 6-11 Hovering Over an Oracle CQL Operator



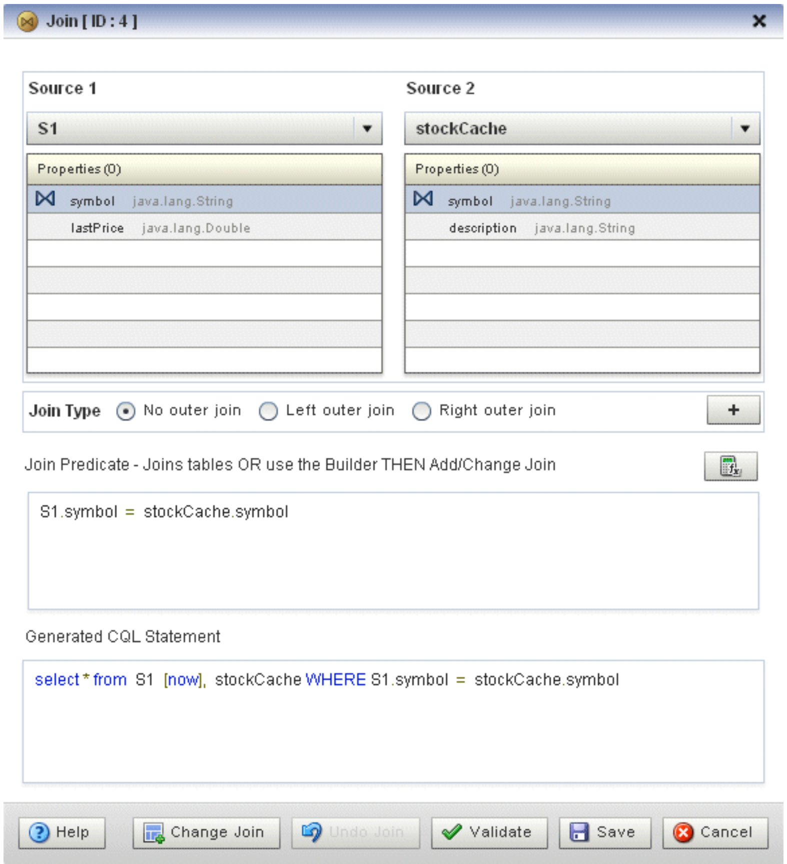
* from Merged AS a

7. To modify the Oracle CQL statements associated with any operator, double click the operator.

   The corresponding editor dialog appears.

   Figure 6-12 shows the editor dialog after double clicking the **Join** operator.

Figure 6-12 Editor Dialog for Oracle CQL Join Operator



Join [ID : 4]

Source 1: S1

Properties (0)	
⋈ symbol	java.lang.String
lastPrice	java.lang.Double

Source 2: stockCache

Properties (0)	
⋈ symbol	java.lang.String
description	java.lang.String

Join Type ● No outer join ○ Left outer join ○ Right outer join

Join Predicate - Joins tables OR use the Builder THEN Add/Change Join

```
S1.symbol = stockCache.symbol
```

Generated CQL Statement

```
select * from S1 [now], stockCache WHERE S1.symbol = stockCache.symbol
```

8. Use the editor dialog to modify the operator.

   For information on configuring all the Query Wizard editor dialogs, see Configure Oracle CQL Constructs with Query Wizard.

9. Click the **Add Join** button to apply your changes to the generated Oracle CQL statement.

10. Click the **Validate** button to ensure that your changes are syntactically correct.

    Correct any errors before proceeding.

11. Click **Save** to save your changes.

    Your query is saved in memory.

12. Double click your query's Output operator.

13. Click **Replace Rule**.

    The Oracle CQL rule is now changed and the new version appears in the CQL Rules tab.

14. Close the Output construct.

6.1.3 Create a Parameterized Rule

You can create a parameterized Oracle CQL query.

Note:

You cannot delete a parameterized query and you cannot edit the Oracle CQL query using the Query Wizard.

For more information, see:

- Edit a Parameterized Query
- *Developing Applications for Oracle Event Processing*.

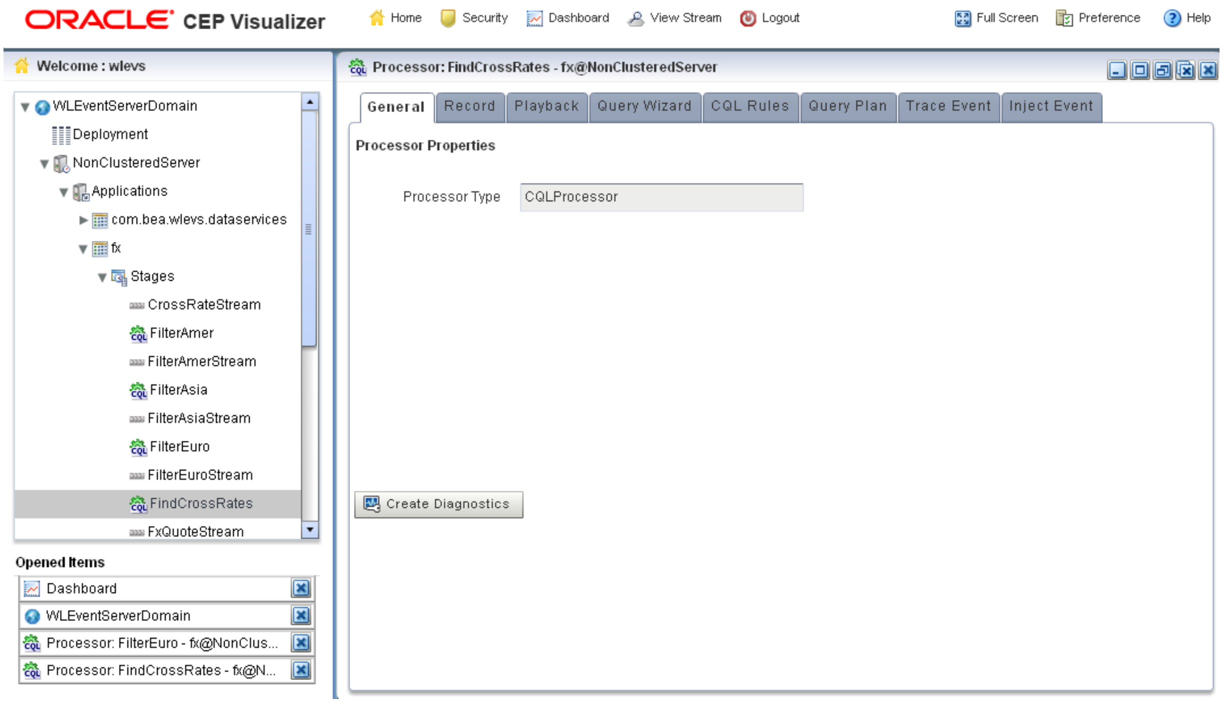
Create a Parameterized Rule

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
    a. To use the EPN diagram:
        - Click the **Event Processing Network** tab.
        - Double-click the Oracle CQL processor you wish to use.

    b. To use the domain tree:
        - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
        - Click the Oracle CQL processor you wish to use.
        - In the right pane, click the **General** tab

    The Processor panel is displayed as Figure 6-13 shows.

Figure 6-13 Oracle CQL Processor Panel



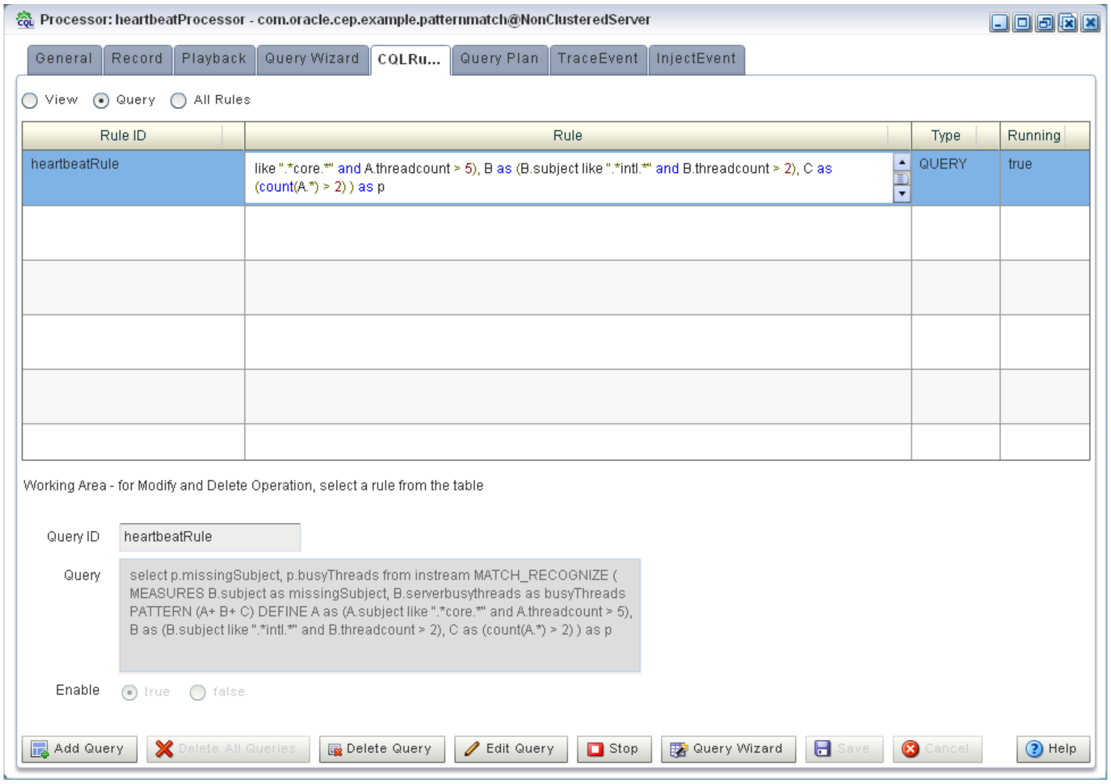
Processor Properties

Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-14 shows.

Figure 6-14 CQL Rules Tab



○ View ● Query ○ All Rules

Rule ID	Rule	Type	Running
heartbeatRule	like ".*core.*" and A.threadcount > 5), B as (B.subject like ".*intl.*" and B.threadcount > 2), C as (count(A.*) > 2)) as p	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Query ID: heartbeatRule

Query: select p.missingSubject, p.busyThreads from instream MATCH_RECOGNIZE (MEASURES B.subject as missingSubject, B.serverbusythreads as busyThreads PATTERN (A+ B+ C) DEFINE A as (A.subject like ".*core.*" and A.threadcount > 5), B as (B.subject like ".*intl.*" and B.threadcount > 2), C as (count(A.*) > 2)) as p

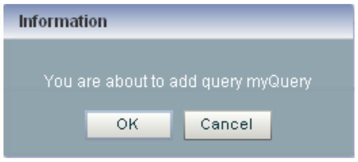
Enable: ● true ○ false

5. Click the **Query** radio button.
6. Click **Add Query**.
7. Enter a query name in the **Query Id** field.
8. Enter the Oracle CQL query in the **Query** field.

   Use placeholder characters (:*n*) to specify parameterized values.
9. Configure the **Enable** option:
   a. To enable the query as soon as it is saved, click **true**.
   b. To keep the query disabled after it is saved, click **false**.
10. Click **Save**.

    A confirmation dialog appears as Figure 6-15 shows.

    Figure 6-15 Query Save Dialog

    Information

    You are about to add query myQuery

11. Click **OK**.

    Your query is saved to the local disk of the computer you are currently using. To reload your query, you must access the Oracle Event Processing Visualizer from the same local host.
12. Edit the parameterized query to add parameter values as Edit a Parameterized Query describes.
13. After you create a query, update the selector for the outbound channel (the channel down-stream from the Oracle CQL processor you created your query on). This will only work with pre-existing event types.

    For more information, see Channel Properties: Outbound Channel .

6.1.4 Edit a Parameterized Query

You can edit the parameter values of an existing Oracle CQL parameterized query. You can add a new list of parameter values, modify an existing list of parameter values, or delete an existing list of parameter values.

Note:

You cannot add or remove placeholder characters (:*n*) nor can you edit the Oracle CQL query using the Query Wizard.

For more information, see:

- Create a Parameterized Rule
- *Oracle CQL Language Reference for Oracle Event Processing*.

Edit a Parameterized Query

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application you want to use.

3. Select the Oracle CQL processor you wish to use:

   a. To use the EPN diagram:

      - Click the **Event Processing Network** tab.

      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:

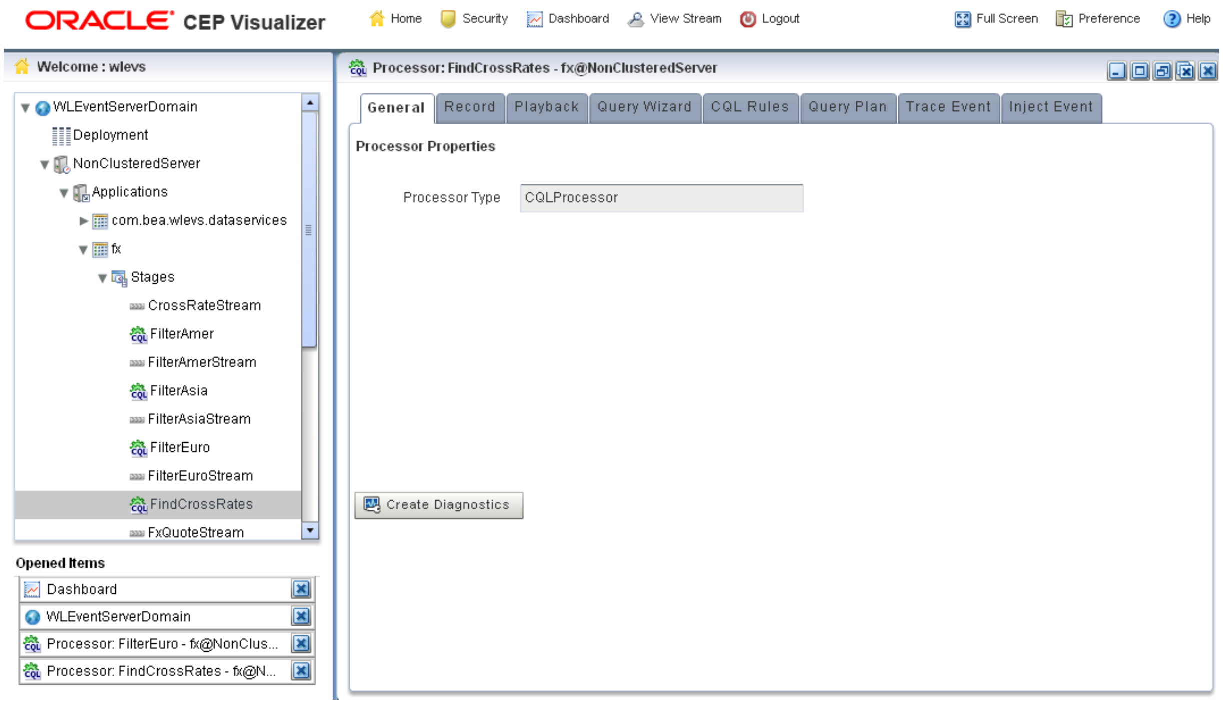
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.

      - Click the Oracle CQL processor you wish to use.

      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-16 shows.

Figure 6-16 Oracle CQL Processor Panel

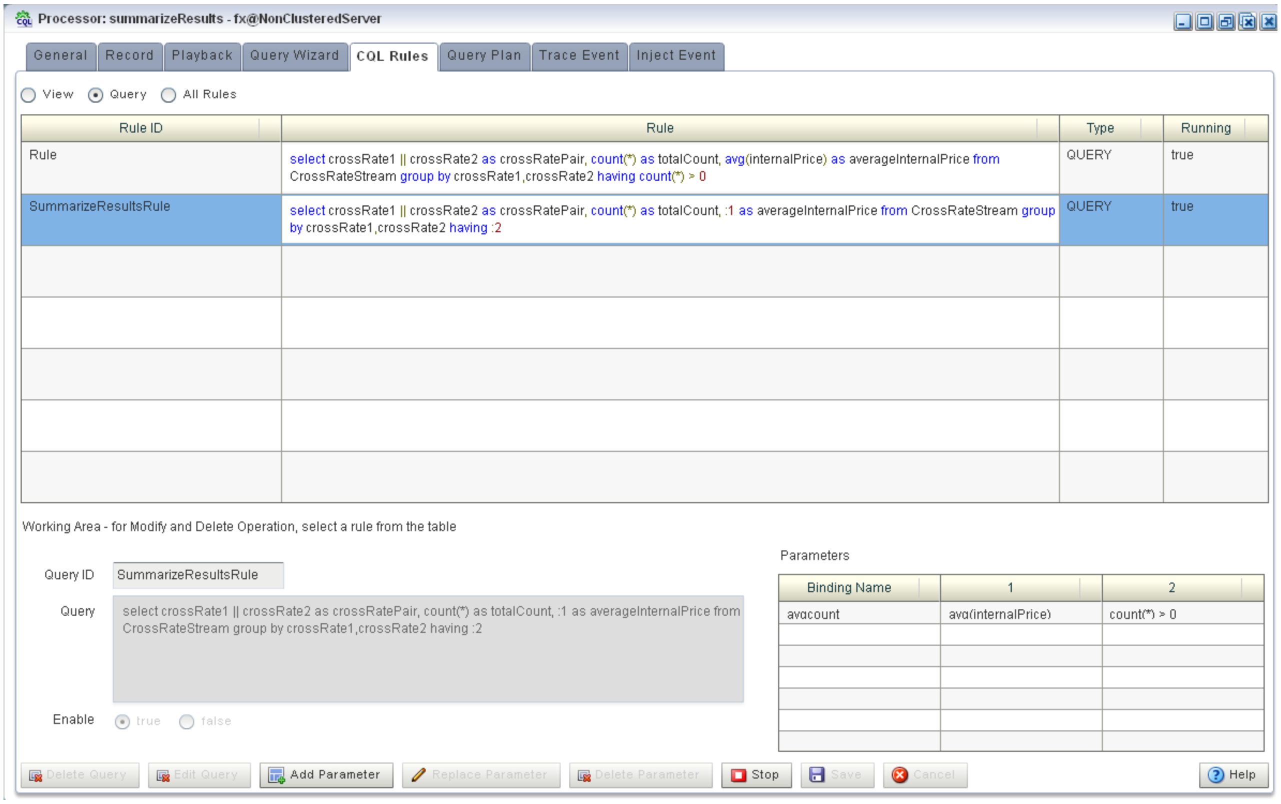


Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-17 shows.

Figure 6-17 CQL Rules Tab: Parameterized Query



Rule ID	Rule	Type	Running
Rule	select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, avg(internalPrice) as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having count(*) > 0	QUERY	true
SummarizeResultsRule	select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Query ID: SummarizeResultsRule

Query: select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2

Enable ● true ○ false

Parameters

Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0

Note:

The Query Wizard button is not available for parameterized queries.

5. Select the parameterized query.

6. Use the CQL Rules tab to modify the paramaterized query:

   - Add a Parameter
   - Replace a Parameter
   - Delete a Parameter
   - Edit the Query
   - Delete the Query

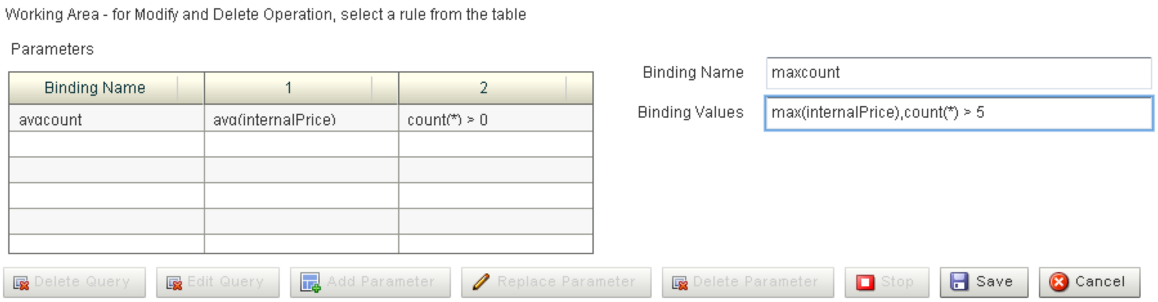
6.1.4.1 Add a Parameter

To add a parameter to a parameterized query:

1. Click the **Add Parameter** button.

   The CQL Rules tab changes as Figure 6-18 shows.

Figure 6-18 CQL Rules Tab: Add Parameter



Working Area - for Modify and Delete Operation, select a rule from the table

Parameters

Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0

Binding Name: maxcount

Binding Values: max(internalPrice),count(*) > 5

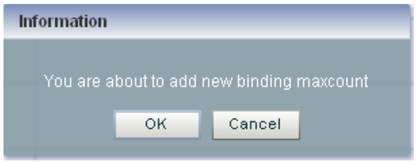
2. Enter a binding name in the **Binding Name** field.

3. Enter the new parameter values as a comma-separated list of values in the **Binding Values** field.

4. Click **Save**.

   The Add New Binding Dialog appears as Figure 6-19 shows.

   Figure 6-19 Add New Binding Dialog

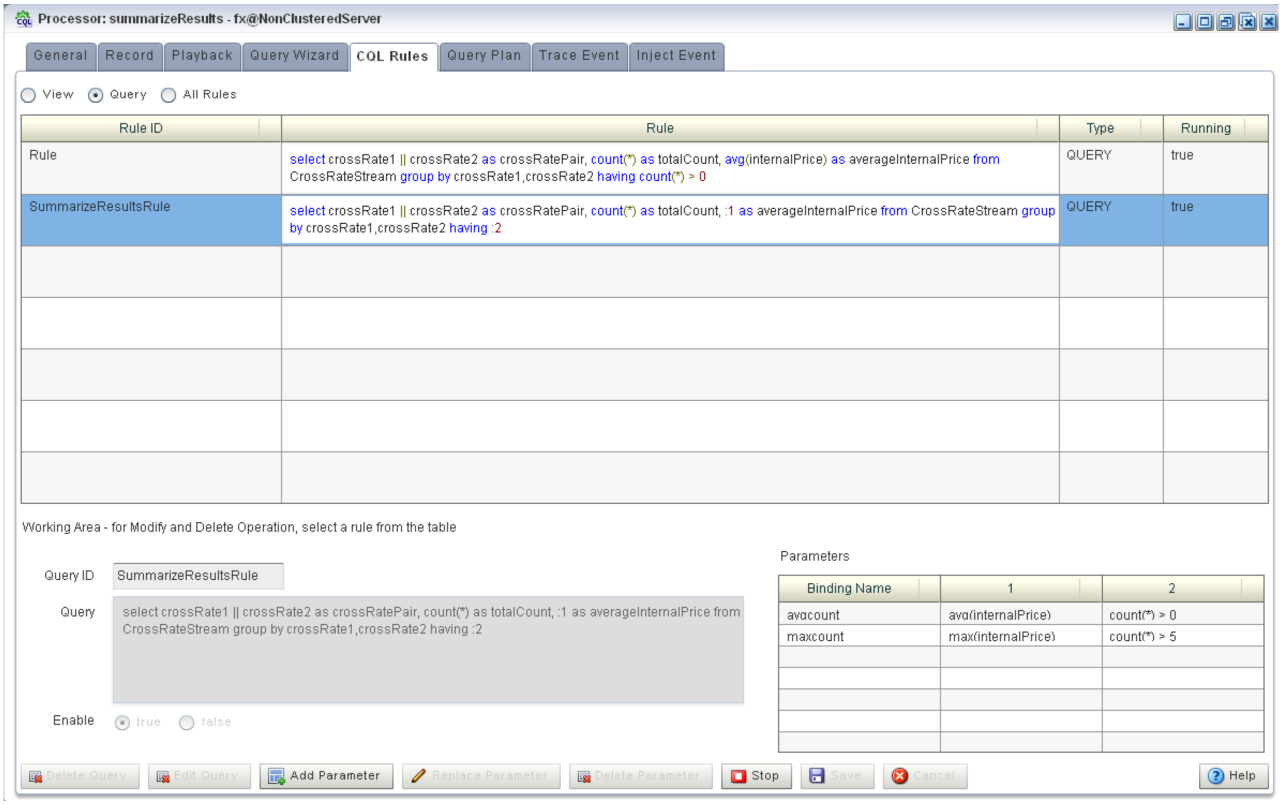
   Information

   You are about to add new binding maxcount

5. Click **OK**.

   The CQL Rules tab updates to list the new parameter as Figure 6-20 shows.

Figure 6-20 CQL Rules Tab: Add Parameter Complete



Rule ID	Rule	Type	Running
Rule	select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, avg(internalPrice) as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having count(*) > 0	QUERY	true
SummarizeResultsRule	select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Query ID: SummarizeResultsRule

Query: select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2

Enable ● true ○ false

Parameters

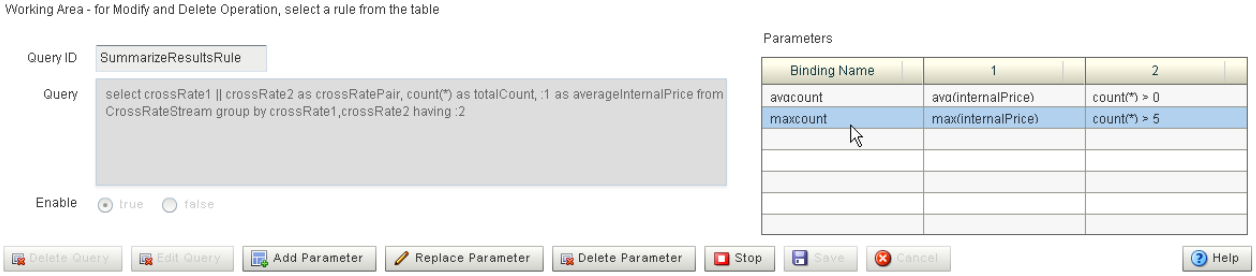
Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0
maxcount	max(internalPrice)	count(*) > 5

6.1.4.2 Replace a Parameter

To replace a parameter in a paramaterized query:

1. Select an existing parameter as Figure 6-21 shows.

Figure 6-21 CQL Rules Tab: Select Parameter



Working Area - for Modify and Delete Operation, select a rule from the table

Query ID: SummarizeResultsRule

Query: select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2

Enable ● true ○ false

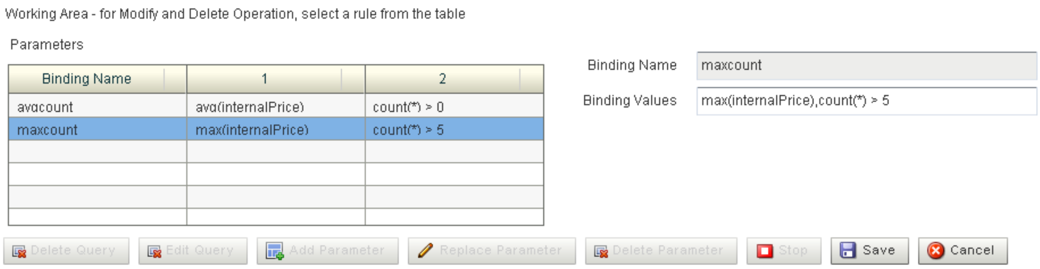
Parameters

Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0
maxcount	max(internalPrice)	count(*) > 5

2. Click **Replace Parameter**.

3. The CQL Rules tab changes as Figure 6-22 shows.

Figure 6-22 CQL Rules: Replace Parameter



Working Area - for Modify and Delete Operation, select a rule from the table

Parameters

Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0
maxcount	max(internalPrice)	count(*) > 5

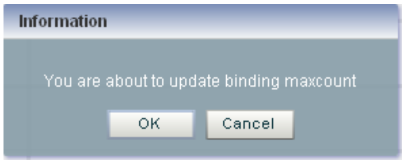
Binding Name: maxcount

Binding Values: max(internalPrice),count(*) > 5

4. Edit the **Binding Values** field.
5. Click **Save**.

   The Update Binding Dialog appears as Figure 6-23 shows.

Figure 6-23 Update Binding Dialog



Information

You are about to update binding maxcount

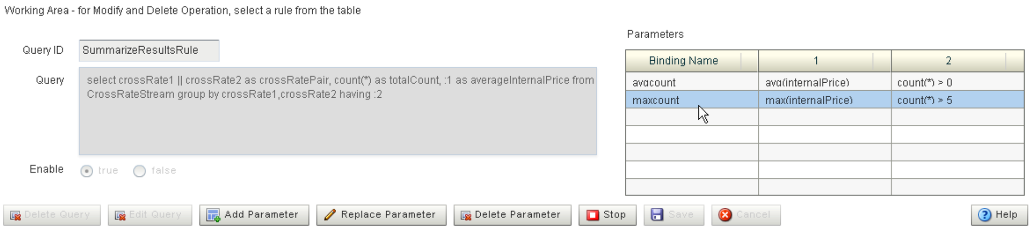
6. Click **OK**.

6.1.4.3 Delete a Parameter

To delete a parameter in a parametrized query:

1. Select an existing parameter as Figure 6-24 shows.

Figure 6-24 CQL Rules Tab: Select Parameter



Working Area - for Modify and Delete Operation, select a rule from the table

Query ID: SummarizeResultsRule

Query: select crossRate1 || crossRate2 as crossRatePair, count(*) as totalCount, :1 as averageInternalPrice from CrossRateStream group by crossRate1,crossRate2 having :2

Enable: ● true ○ false

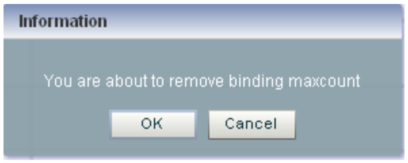
Parameters

Binding Name	1	2
avgcount	avg(internalPrice)	count(*) > 0
maxcount	max(internalPrice)	count(*) > 5

2. Click **Delete Parameter**.

   The Delete Binding Dialog appears as Figure 6-25 shows.

Figure 6-25 Delete Binding Dialog



Information

You are about to remove binding maxcount

3. Click **OK**.

6.1.4.4 Edit the Query

To edit a parameterized query, you must first delete all query parameters before the Edit Query button is available. For information on deleting parameters, see Delete a Parameter.

6.1.4.5 Delete the Query

To delete a parameterized query, you must first delete all query parameters before the Delete Query button is available. For more information, see Delete a Parameter.

6.1.5 Stop and Start a Parameterized Query

When you stop or start a parameterized query, all bindings associated with the parameterized query are stopped or started.

To stop or start a parameterized query, select the query on the CQL Rules tab and click the **Stop** or **Start** button.

6.1.6 Delete a Rule in an Oracle CQL Processor

You can delete an existing Oracle CQL view or query or delete all Oracle CQL views or queries in an Oracle CQL processor.

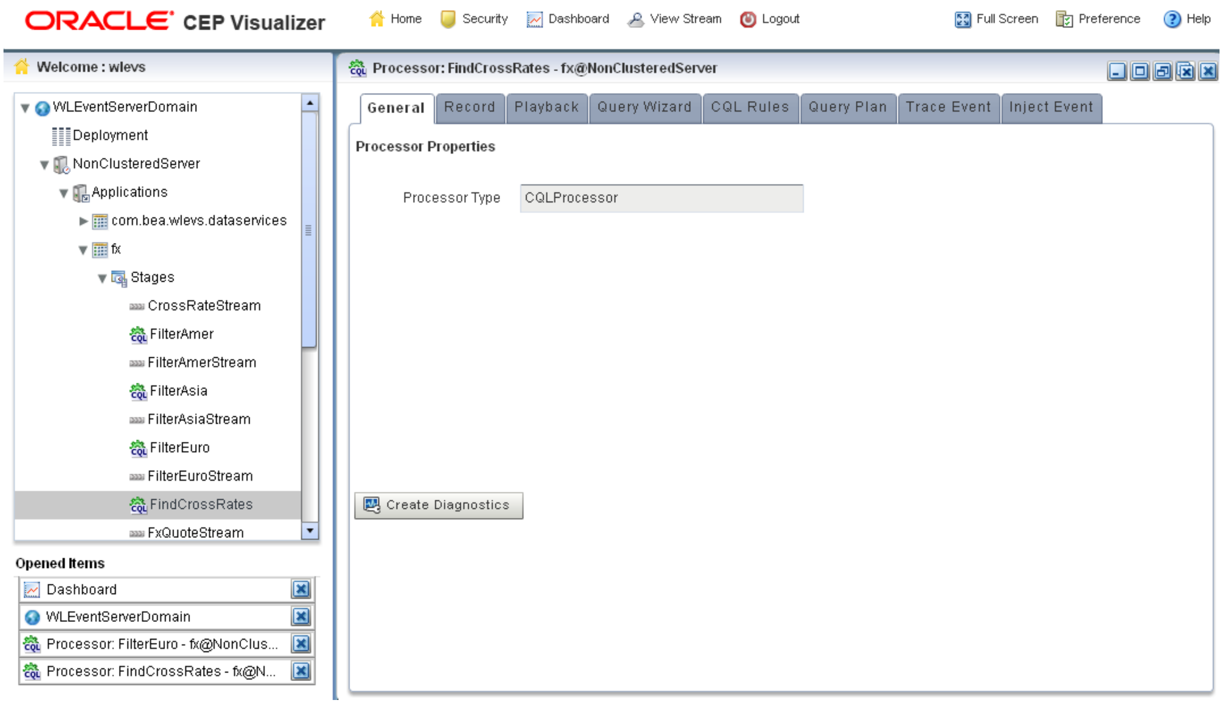
Delete a Rule

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-26 shows.

Figure 6-26 Oracle CQL Processor Panel



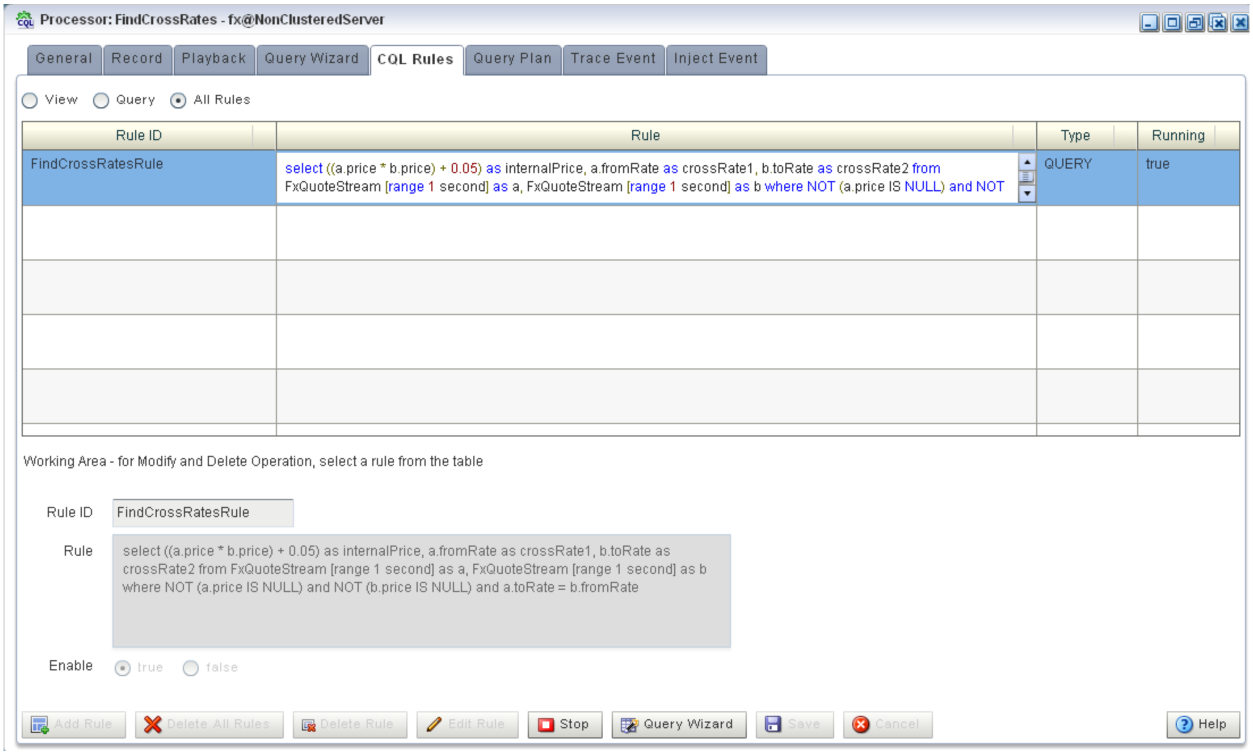
Processor Properties

Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-27 shows.

Figure 6-27 CQL Rules Tab



○ View ○ Query ● All Rules

Rule ID	Rule	Type	Running
FindCrossRatesRule	select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Rule ID: FindCrossRatesRule

Rule: select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT (b.price IS NULL) and a.toRate = b.fromRate

Enable: ● true ○ false

5. Filter the list of rules to show only the type of rule you want to delete:

a. Select **View** to see only view rules.

b. Select **Query** to see only query views.

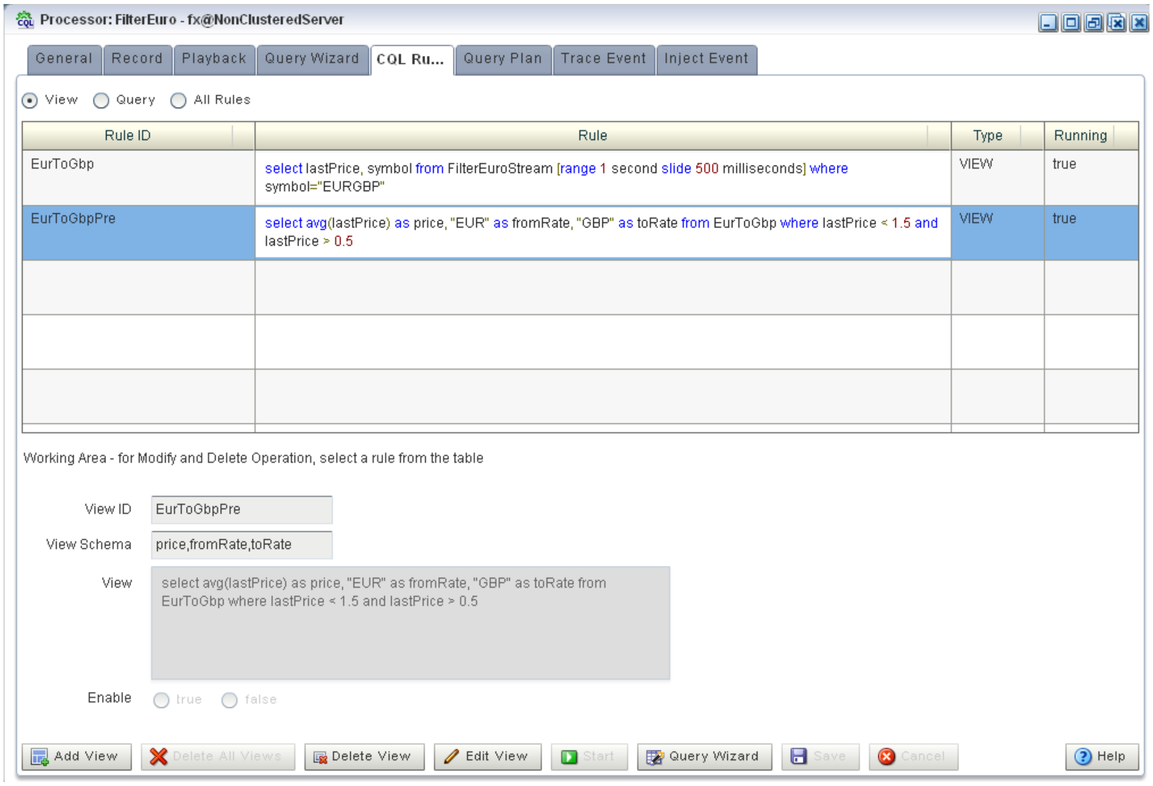
Note:

You cannot add, delete, or modify a single rule if you select **All Rules**. You must select a particular view or query to add, delete, or modify a single rule. To make changes to either views or queries, you must select **View** or **Query**. The **All Rules** page is a read-only panel.

Also note that you can turn off a query dynamically using the **Enable** radio buttons.

The Delete All Views or Delete All Queries button is active as Figure 6-28 shows.

Figure 6-28 CQL Rules: Filtered by View



● View ○ Query ○ All Rules

Rule ID	Rule	Type	Running
EurToGbp	select lastPrice, symbol from FilterEuroStream [range 1 second slide 500 milliseconds] where symbol="EURGBP"	VIEW	true
EurToGbpPre	select avg(lastPrice) as price, "EUR" as fromRate, "GBP" as toRate from EurToGbp where lastPrice < 1.5 and lastPrice > 0.5	VIEW	true

Working Area - for Modify and Delete Operation, select a rule from the table

View ID: EurToGbpPre

View Schema: price,fromRate,toRate

View: select avg(lastPrice) as price, "EUR" as fromRate, "GBP" as toRate from EurToGbp where lastPrice < 1.5 and lastPrice > 0.5

Enable ○ true ○ false

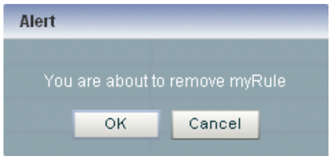
6. Decide what rules you want to delete:

a. To delete a single rule:

- Select the rule you want to delete.
- Click **Delete View** or **Delete Query**.

  A confirmation dialog appears as Figure 6-29 shows.

Figure 6-29 Delete Rule Dialog



Alert

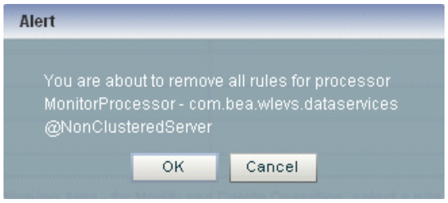
You are about to remove myRule

b. To delete all rules:

- Click **Cancel** to unselect any rules that may be currently selected.
- Click **Delete All Views** or **Delete All Queries**.

  A confirmation dialog appears as Figure 6-30 shows.

Figure 6-30 Delete All Rule Dialog



Alert

You are about to remove all rules for processor MonitorProcessor - com.bea.wlevs.dataservices @NonClusteredServer

7. Click **OK**.

6.1.7 Replace a Rule

Using Oracle Event Processing Visualizer, you can replace an existing view or query by either editing the Oracle CQL statement directly or using the Query Wizard.

Replace a Rule

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:

   a. To use the EPN diagram:

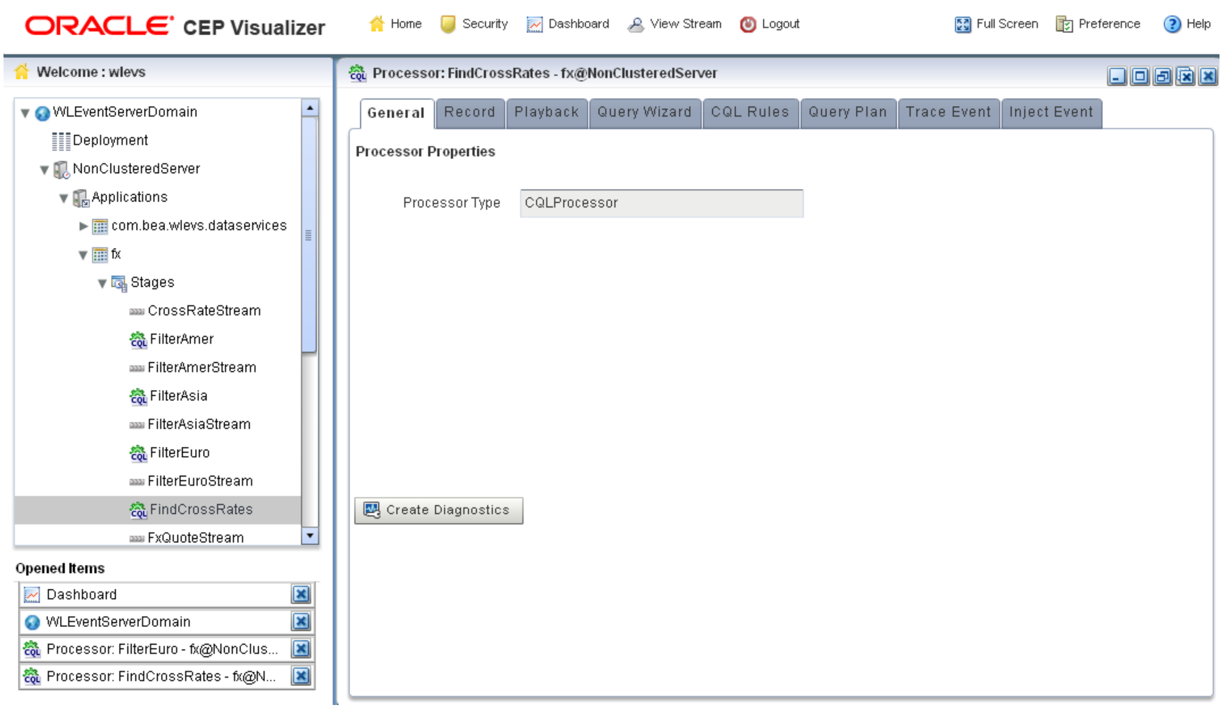
   - Click the **Event Processing Network** tab.
   - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:

   - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
   - Click the Oracle CQL processor you wish to use.
   - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-31 shows.

Figure 6-31 Oracle CQL Processor Panel



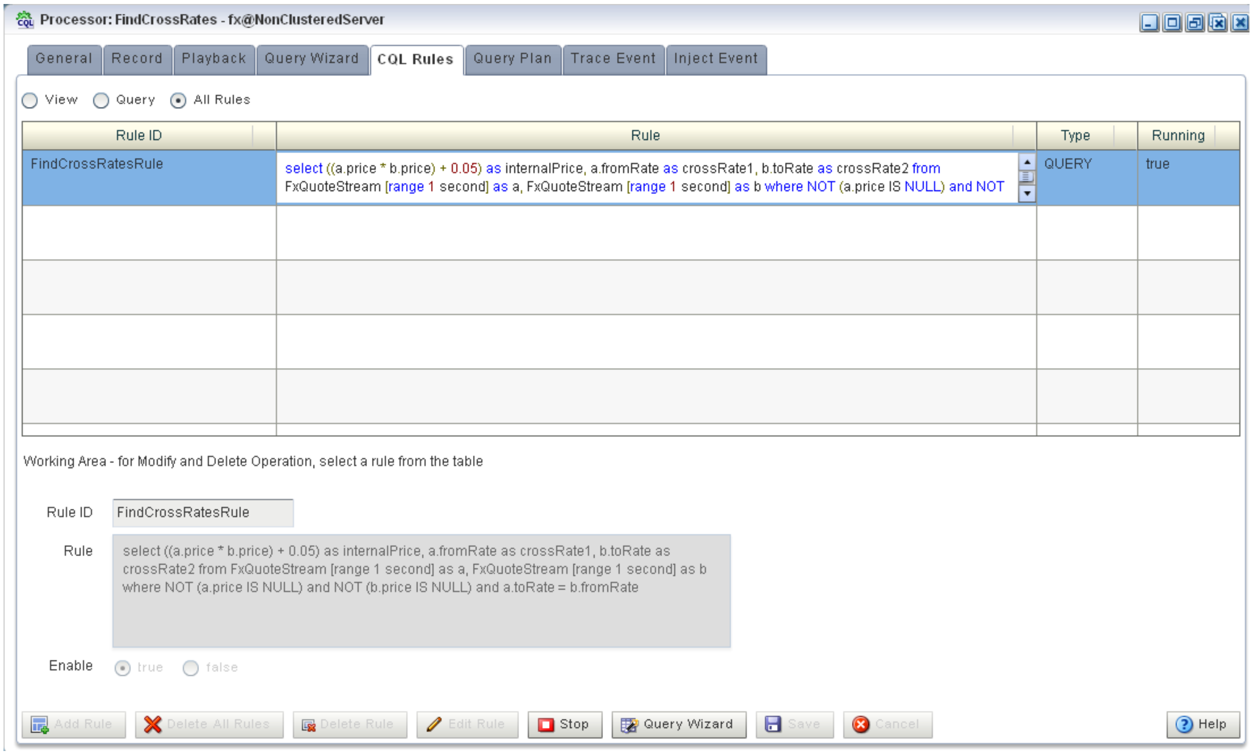
Processor Properties

Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-32 shows.

Figure 6-32 CQL Rules Tab



○ View ○ Query ● All Rules

Rule ID	Rule	Type	Running
FindCrossRatesRule	select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Rule ID: FindCrossRatesRule

Rule: select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT (b.price IS NULL) and a.toRate = b.fromRate

Enable: ● true ○ false

5. Filter the list of rules to show only the type of rule you want to replace:

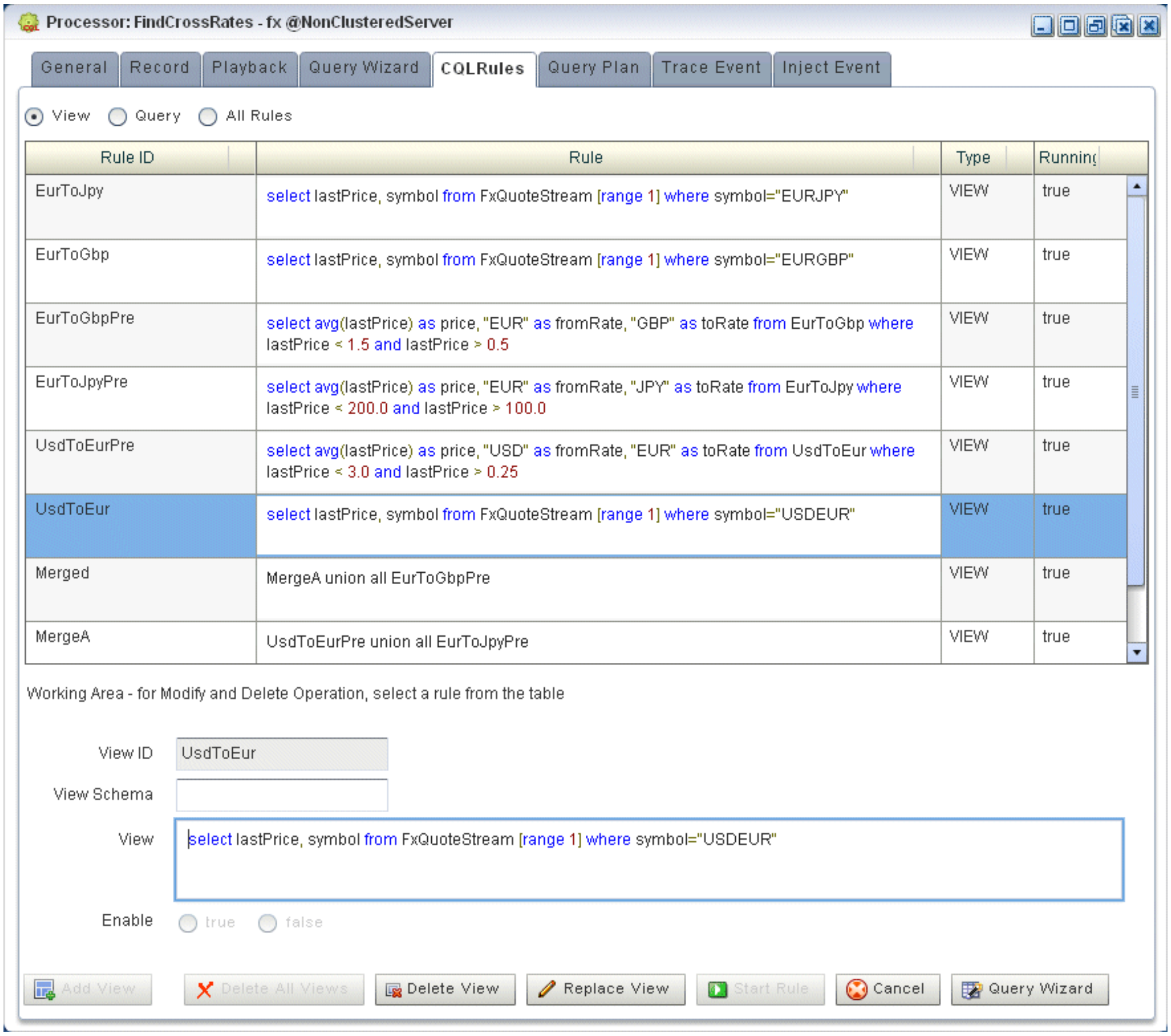
a. Select **View** to see only view rules.

b. Select **Query** to see only query views.

6. Select the rule you wish to replace.

   The View or Query field is now editable and the Replace Rule button is active as Figure 6-33 shows.

Figure 6-33 CQL Rules: Filtered by View



● View ○ Query ○ All Rules

Rule ID	Rule	Type	Running
EurToJpy	select lastPrice, symbol from FxQuoteStream [range 1] where symbol="EURJPY"	VIEW	true
EurToGbp	select lastPrice, symbol from FxQuoteStream [range 1] where symbol="EURGBP"	VIEW	true
EurToGbpPre	select avg(lastPrice) as price, "EUR" as fromRate, "GBP" as toRate from EurToGbp where lastPrice < 1.5 and lastPrice > 0.5	VIEW	true
EurToJpyPre	select avg(lastPrice) as price, "EUR" as fromRate, "JPY" as toRate from EurToJpy where lastPrice < 200.0 and lastPrice > 100.0	VIEW	true
UsdToEurPre	select avg(lastPrice) as price, "USD" as fromRate, "EUR" as toRate from UsdToEur where lastPrice < 3.0 and lastPrice > 0.25	VIEW	true
UsdToEur	select lastPrice, symbol from FxQuoteStream [range 1] where symbol="USDEUR"	VIEW	true
Merged	MergeA union all EurToGbpPre	VIEW	true
MergeA	UsdToEurPre union all EurToJpyPre	VIEW	true

Working Area - for Modify and Delete Operation, select a rule from the table

View ID: UsdToEur

View Schema:

View: select lastPrice, symbol from FxQuoteStream [range 1] where symbol="USDEUR"

Enable: ○ true ○ false

7. Edit the Oracle CQL statement by doing one of the following:

   a. Directly edit the Oracle CQL statement in the **View** or **Query** field.

   b. Click the **Query Wizard** button to edit the rule in the Query Wizard.

      For more information, see Edit a Rule with the Query Wizard.

8. Commit your changes:

   a. If you directly edited the Oracle CQL statement in the **View** or **Query** field, click **Replace View** or **Replace Query**.

b. If you edited the Oracle CQL statement using the Query Wizard, you commit your changes in the Query Wizard.

   For more information, see Edit a Rule with the Query Wizard.

6.1.8 View a Rule

Using Oracle Event Processing Visualizer, you can view the existing rules defined on an Oracle CQL processor.

View a Rule

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:

   a. To use the EPN diagram:

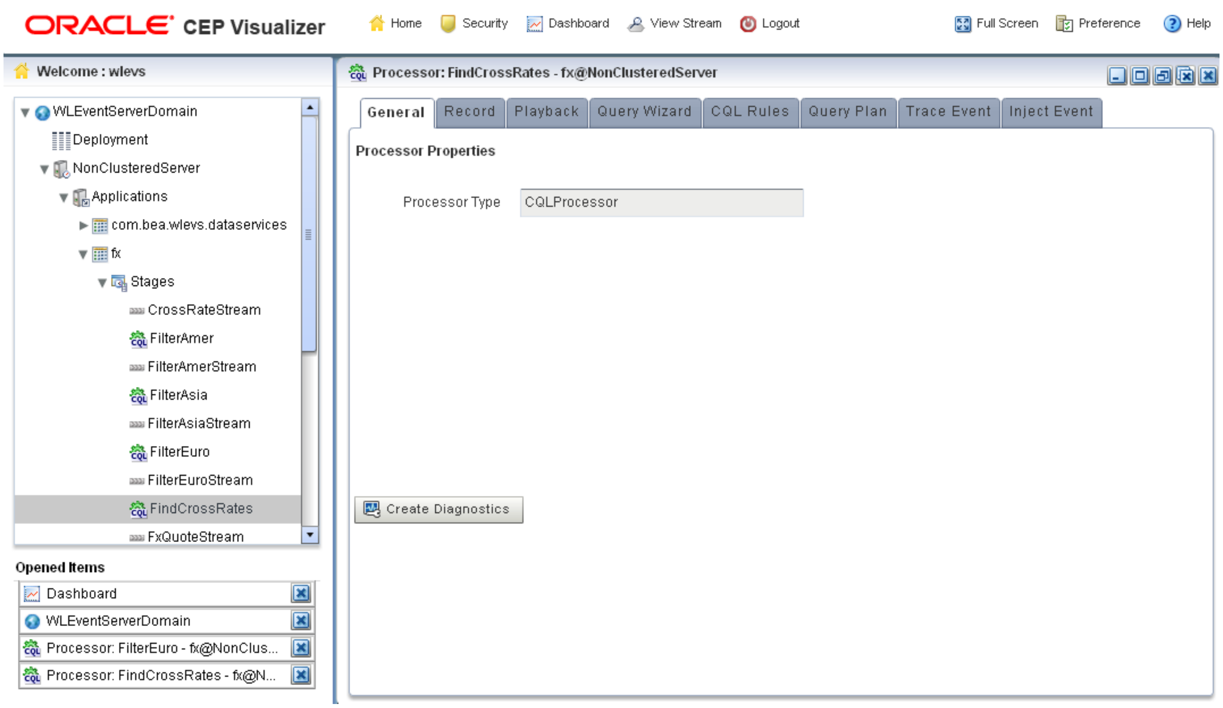
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:

      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-34 shows.

Figure 6-34 Oracle CQL Processor Panel

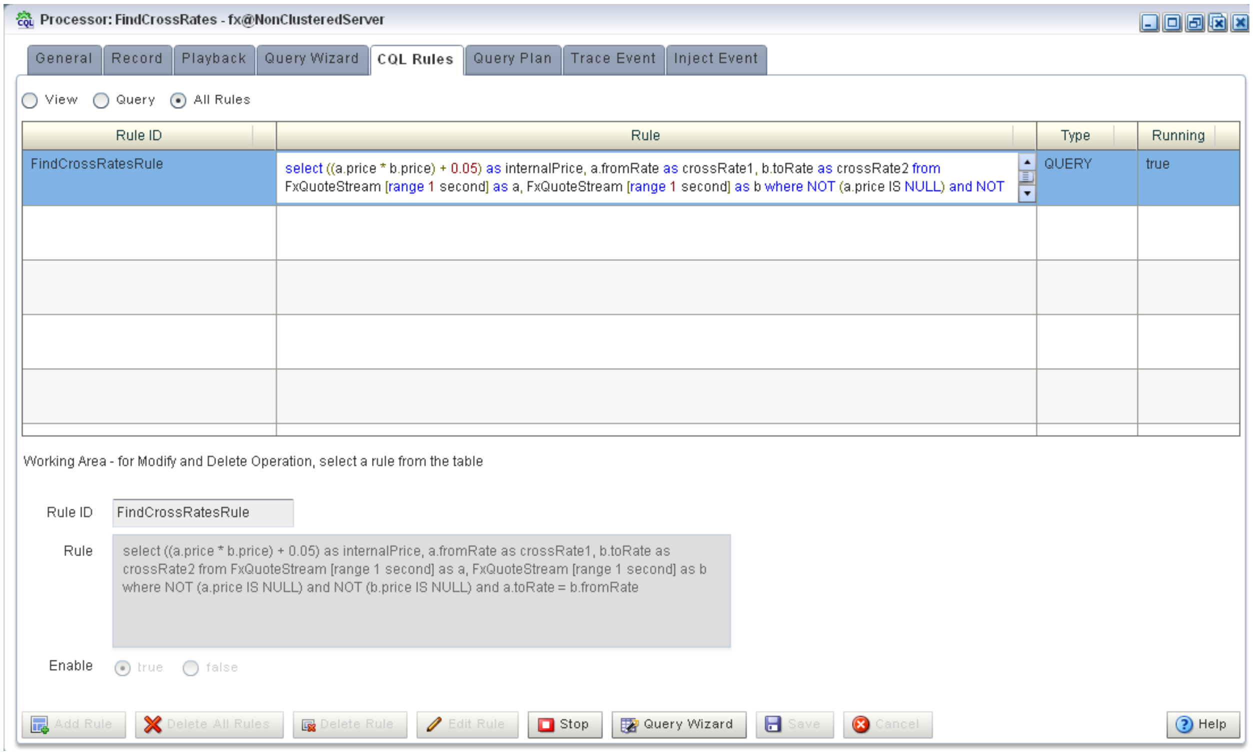


Processor Type: CQLProcessor

4. Click the **CQL Rules** tab.

   The CQL Rules tab appears as Figure 6-35 shows.

Figure 6-35 CQL Rules Tab



○ View ○ Query ● All Rules

Rule ID	Rule	Type	Running
FindCrossRatesRule	select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT	QUERY	true

Working Area - for Modify and Delete Operation, select a rule from the table

Rule ID: FindCrossRatesRule

Rule: select ((a.price * b.price) + 0.05) as internalPrice, a.fromRate as crossRate1, b.toRate as crossRate2 from FxQuoteStream [range 1 second] as a, FxQuoteStream [range 1 second] as b where NOT (a.price IS NULL) and NOT (b.price IS NULL) and a.toRate = b.fromRate

Enable: ● true ○ false

6.1.9 Configure Oracle CQL Constructs with Query Wizard

This section describes how to configure the various Oracle CQL constructs that the Query Wizard provides, including:

- Configure an RSource CQL Construct
- Configure an SSource CQL Construct
- Configure a Cache-Table CQL Construct
- Configure a Pattern CQL Construct
- Configure a Select CQL Construct
- Configure a Join CQL Construct Using + Syntax
- Configure a Join CQL Construct Using LEFT or RIGHT OUTER JOIN
- Configure a Window CQL Construct
- Configure a Filter CQL Construct
- Configure a Union CQL Construct
- Configure an Intersect CQL Construct
- Configure a Minus CQL Construct
- Configure an IStream CQL Construct
- Configure a DStream CQL Construct
- Configure an RStream CQL Construct
- Configure an Output CQL Construct
- Configure an Expression Using the Expression Builder

6.1.9.1 Configure an RSource CQL Construct

The RSource Oracle CQL construct represents an Oracle CQL relation that your Oracle Event Processing application provides or a relation type of view that you create. To create a stream type of source, an SSource, see Configure an SSource CQL Construct.

For more information and detailed syntax, see Oracle Fusion Middleware Oracle CQL Language Reference for Oracle Event Processing.

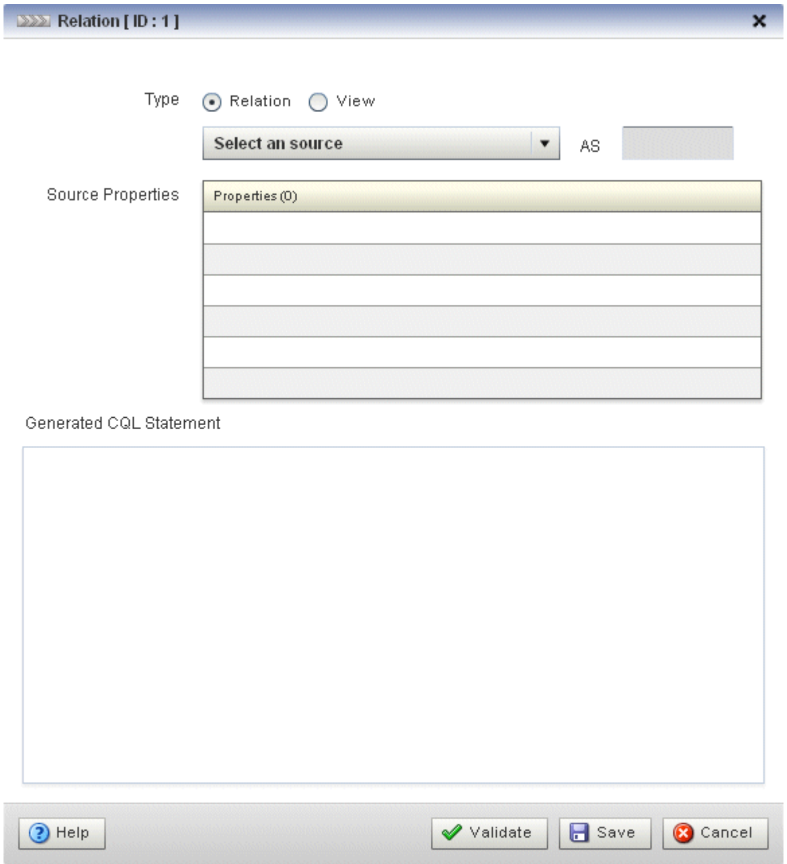
Configure an RSource CQL Construct

1. Double click the **RSource** icon as Figure 6-36 shows.

   Figure 6-36 RSource Icon

   The RSource editor dialog appears as Figure 6-37 shows.

Figure 6-37 Query Wizard RSource Editor Dialog



2. Edit the dialog as Table 6-1 describes.

Table 6-1 Query Wizard RSource Editor Attributes

Attribute	Description
`Type`	Select the type of source: • **Relation**: select this option to process events offered by a relation (query) currently running on this Oracle CQL processor. • **View**: select this option to process events offered by a view currently running on this Oracle CQL processor.
`Select a source`	Select the relation or view to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

The RSource icon displays a green check mark to indicate that it is configured.

6.1.9.2 Configure an SSource CQL Construct

The SSource Oracle CQL construct represents an Oracle CQL stream that your Oracle Event Processing application provides or a stream type of view you create. To create a relation type of source, an RSource, see Configure an RSource CQL Construct. See *Oracle CQL Language Reference for Oracle Event Processing*.

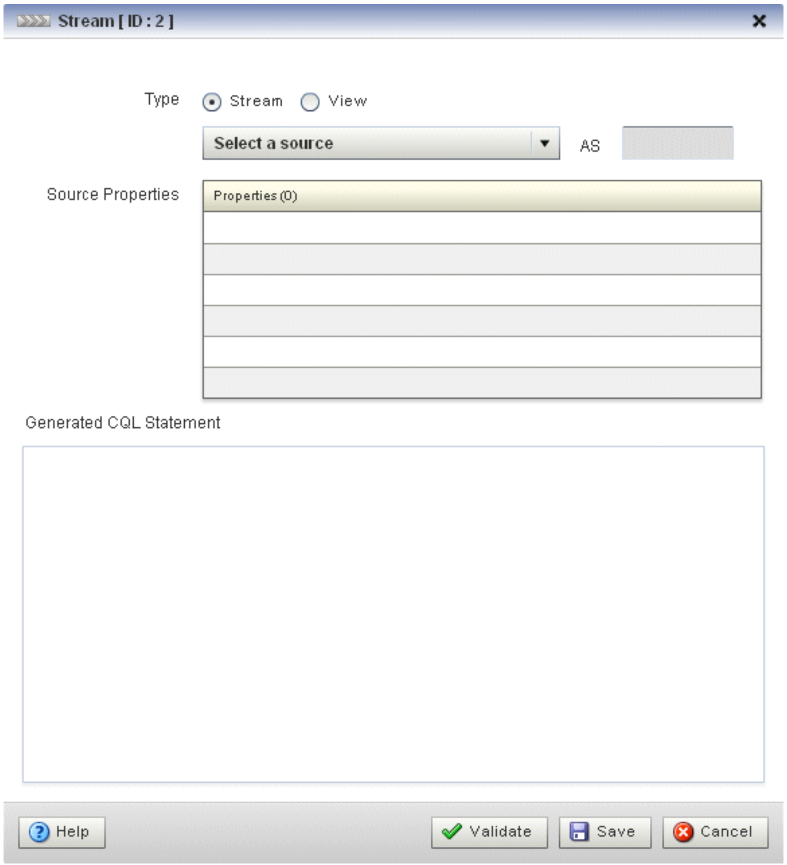
Configure an SSource CQL Construct

1. Double click the **SSource** icon as Figure 6-38 shows.

Figure 6-38 SSource Icon

The SSource editor dialog appears as Figure 6-39 shows.

Figure 6-39 Query Wizard Source Editor Dialog



Stream [ID : 2]

Type ● Stream ○ View

Select a source ▾ AS

Source Properties

Properties (0)

Generated CQL Statement

2. Edit the dialog as Table 6-2 describes.

Table 6-2 Query Wizard Source Editor Attributes

Attribute	Description
`Type`	Select the type of source: • **Stream**: select this option to process events offered by a stream provided by your Oracle Event Processing application. • **View**: select this option to process events offered by a view currently running on this Oracle CQL processor.
`Select a source`	Select the stream or view to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The Source icon displays a green check mark to indicate that it is configured.

6.1.9.3 Configure a Cache-Table CQL Construct

The Cache-Table CQL construct represents an external relation, usually a pull data source, that a cache or relational database table provides. The only operation that is allowed on the Cache-Table CQL construct is a join with another stream source using the `Now` window.

For more information, see:

- Configure a Join CQL Construct Using + Syntax
- Configure a Window CQL Construct

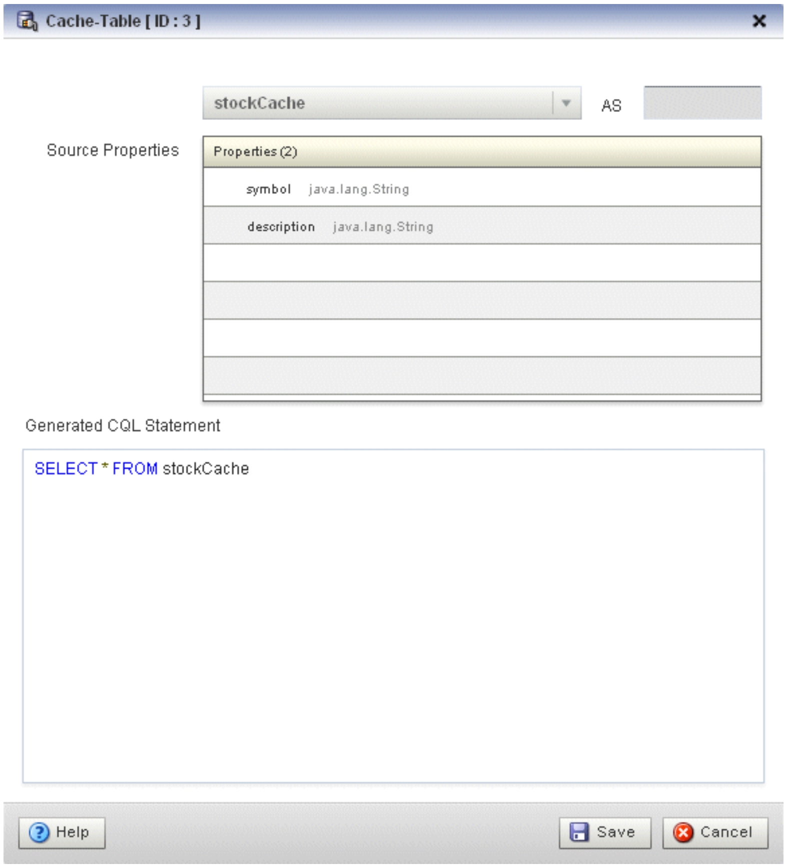
Configure a Cache-Table CQL Construct

1. Double click the **Cache-Table** icon as Figure 6-40 shows.

   Figure 6-40 Cache-Table Icon

   The Cache-Table editor dialog appears as Figure 6-41 shows.

Figure 6-41 Query Wizard Cache-Table Editor Dialog



```
SELECT * FROM stockCache
```

2. Edit the dialog as Table 6-3 describes.

Table 6-3 Query Wizard Cache-Table Editor Attributes

Attribute	Description
`Select a source`	Select the cache to use as the source of events.
`As`	Optionally specify an alias for this source.
`Source Properties`	Read-only list of the properties of the event offered by this source.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click **Save**.

   The Cache-Table icon displays a green check mark to indicate that it is configured.

6.1.9.4 Configure a Pattern CQL Construct

The Pattern Oracle CQL construct represents the Oracle CQL `MATCH_RECOGNIZE` condition. The `MATCH_RECOGNIZE` condition is the principle means of performing event processing. Using its various clauses, you can succinctly express complex conditions among stream elements to perform advanced comparisons optimized for data streams.

The Pattern Oracle CQL construct editor provides a tab for each of the following `MATCH_RECOGNIZE` sub-clauses:

- `PATTERN`: Use this clause to specify quantifiers that constrain and control when conditions in the `DEFINE` clause match.

  See Figure 6-43.

- `DEFINE`: Use this clause to specify one or more conditions. Specify one or more patterns made up of a pattern identifier, the `AS` operator, and a condition. The condition can use any stream elements in the schema of the stream, view, or sliding window specified by the identifier you used in your query or view select statement.

  See Figure 6-44.

- `SUBSET`: Using this clause, you can bind a name to the stream elements that one or more patterns select. You can use this named subset in the `MEASURE` and `DEFINE` clauses.

  See Figure 6-45.

- `MEASURE`: Use this clause to define expressions in a `MATCH_RECOGNIZE` condition and to bind stream elements that match conditions in the `DEFINE` clause to arguments that you can include in the select statement of a query. You can use any of the Oracle CQL built-in or user-defined functions.

  See Figure 6-46.

For more information and detailed syntax, see *Oracle CQL Language Reference for Oracle Event Processing*.

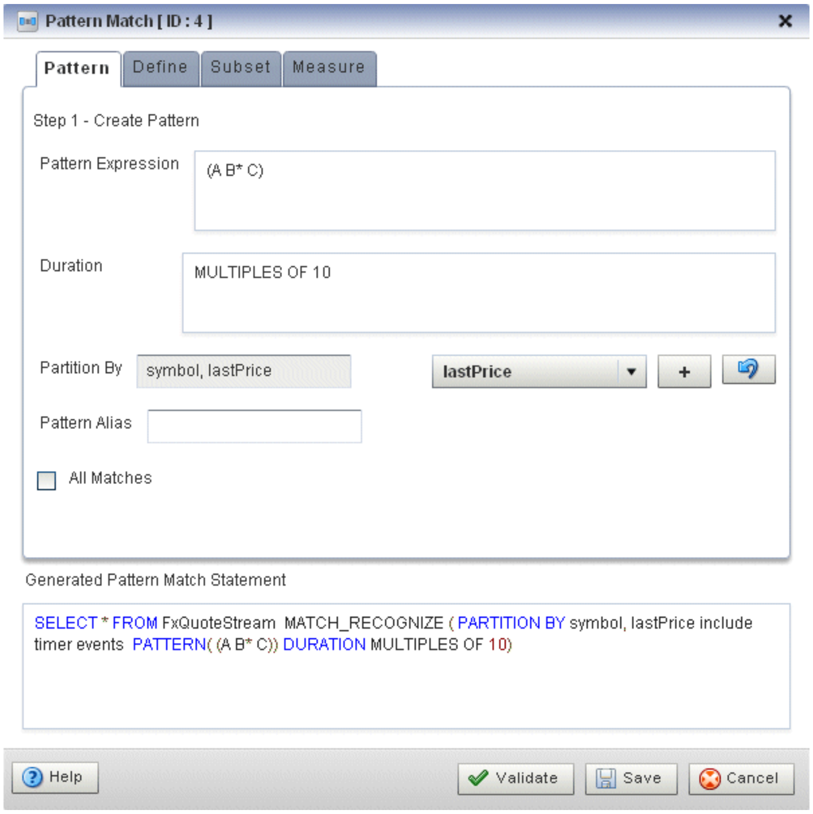
Configure a Pattern CQL Construct

1. Double click the **Pattern** icon as Figure 6-42 shows.

   Figure 6-42 Pattern Icon

   The Pattern editor dialog appears as Figure 6-43 shows.

Figure 6-43 Query Wizard Pattern Editor Dialog: Pattern Tab



Pattern Expression: (A B* C)

Duration: MULTIPLES OF 10

Partition By: symbol, lastPrice — lastPrice

Pattern Alias:

□ All Matches

Generated Pattern Match Statement

```
SELECT * FROM FxQuoteStream  MATCH_RECOGNIZE ( PARTITION BY symbol, lastPrice include
timer events  PATTERN( (A B* C)) DURATION MULTIPLES OF 10)
```

2. Edit the Pattern tab as Table 6-4 describes.

Table 6-4 Query Wizard Pattern Editor Attributes: Pattern Tab

Attribute	Description
`Pattern Expression`	Use this clause to specify quantifiers that constrain and control when conditions in the `DEFINE` clause match. For example: `(A B* C)` Table 6-5 lists the pattern quantifiers Oracle CQL supports. Use the pattern quantifiers to specify the allowed range of pattern matches. The one-character pattern quantifiers are maximal or greedy. They attempt to match the biggest quantity first. The two-character pattern quantifiers are minimal or reluctant. They attempt to match the smallest quantity first.

Attribute	Description
`Duration`	By default, Oracle Event Processing updates the output relation at the time it recognizes a pattern match. Enter a duration value to specify a time delay between when Oracle Event Processing recognizes a pattern match and when it updates the output relation. You can use either of the following: • When you specify a duration like `10`, each time Oracle Event Processing recognizes a pattern match, it delays the specified time interval before updating the output relation. A match happens when the regular expression in the PATTERN clause is matched completely and no other event or input arrives until the duration specified in the DURATION clause expires. The duration is measured from the time of arrival of the first event in the pattern match.There are 'Fixed Duration Non-Event Detection' and 'Recurring Non-Event Detection'. Fixed Duration Non-Event Detection means that the duration can be specified as a constant value, such as 10. Recurring Non-Event Detection means that when you specify a MULTIPLES OF clause, it indicates recurring non-event detection.In this case an output is sent at the multiples of duration value as long as there is no event after the pattern matches completely. • When you specify a duration like `MULTIPLES OF 10`, each time Oracle Event Processing recognizes a pattern match, it delays the specified time interval before updating the output relation. If no input matches the pattern or if no input is received, Oracle Event Processing still updates the output relation. In this case, some terms in the selection may be null and aggregate values may be reset.
`Partition By`	Use this optional clause to specify the stream elements by which a `MATCH_RECOGNIZE` clause should partition its results. You can reference any stream elements in the schema of the stream, view, or sliding window specified by the identifier you used in the query or view select statement. To add a stream element to the partition by list, select a stream element from the pull-down menu and click the **Add** button. To remove a stream element from the partition by list, click the **Undo** button.

`Pattern Alias`	Specifies the `relation_variable` clause `AS` operator to define an alias to label the immediately preceding expression in the select list so that you can reference the result by that name. The alias effectively renames the select list item for the duration of the query.
	Specifying a pattern alias of `its` would correspond to the following example:
	`<query id="detectPerish"><![CDATA[`
	`    select its.itemId as badItem from ItemTempStream`
	`    MATCH_RECOGNIZE (`
	`        PARTITION BY ...`
	`        MEASURES ...`
	`        PATTERN ...`
	`        DEFINE ...`
	`    ) as its`
	`></query>`

Attribute	Description
`All Matches`	Check this option to configure Oracle Event Processing to match overlapping patterns. Uncheck this option to configure Oracle Event Processing to match only one pattern.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

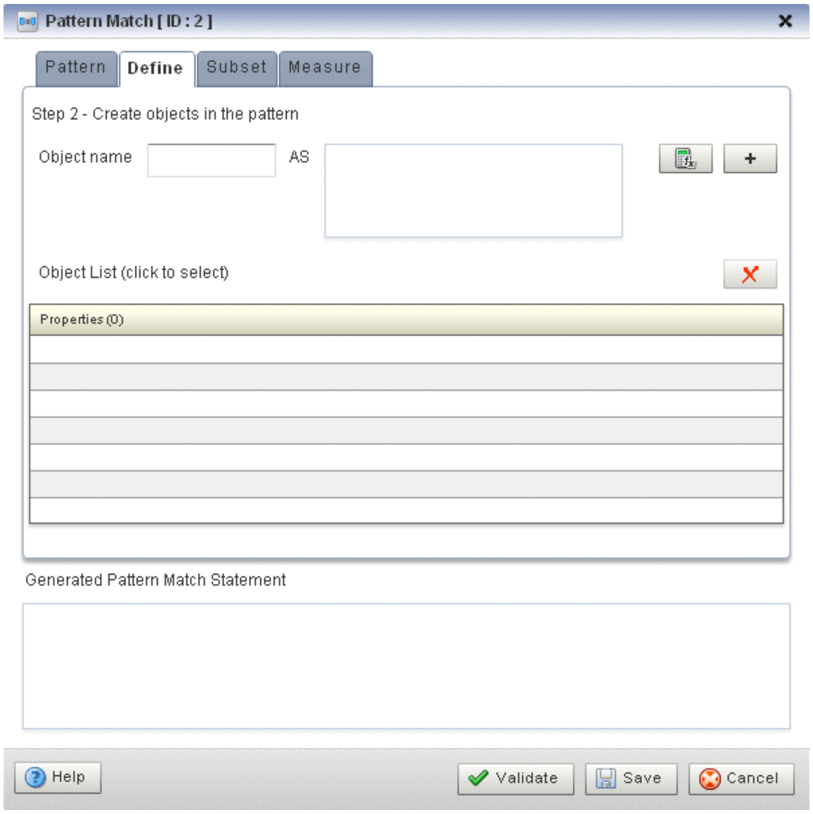
Table 6-5 Pattern Quantifiers

Maximal	Minimal	Description
`*`	`*?`	0 or more times
`+`	`+?`	1 or more times.
`?`	`??`	0 or 1 time.

3. Click the **Define** tab.

   The Define tab appears as Figure 6-44 shows.

Figure 6-44 Query Wizard Pattern Editor Dialog: Define Tab


Pattern Match [ID : 2]
Pattern
Define
Subset
Measure
Step 2 - Create objects in the pattern
Object name
AS
+
Object List (click to select)
Properties (0)
Generated Pattern Match Statement
Help
Validate
Save
Cancel

4. Edit the Define tab as Table 6-6 describes.

Table 6-6 Query Wizard Pattern Editor Attributes: Define Tab

Attribute	Description
`Object Name`	Enter an object name as you specified in the **Pattern Expression** attribute of the Pattern tab (see Table 6-4). For example, if the pattern expression is: `(A B* C)` Then create a definition for each of `A`, `B`, and `C`.

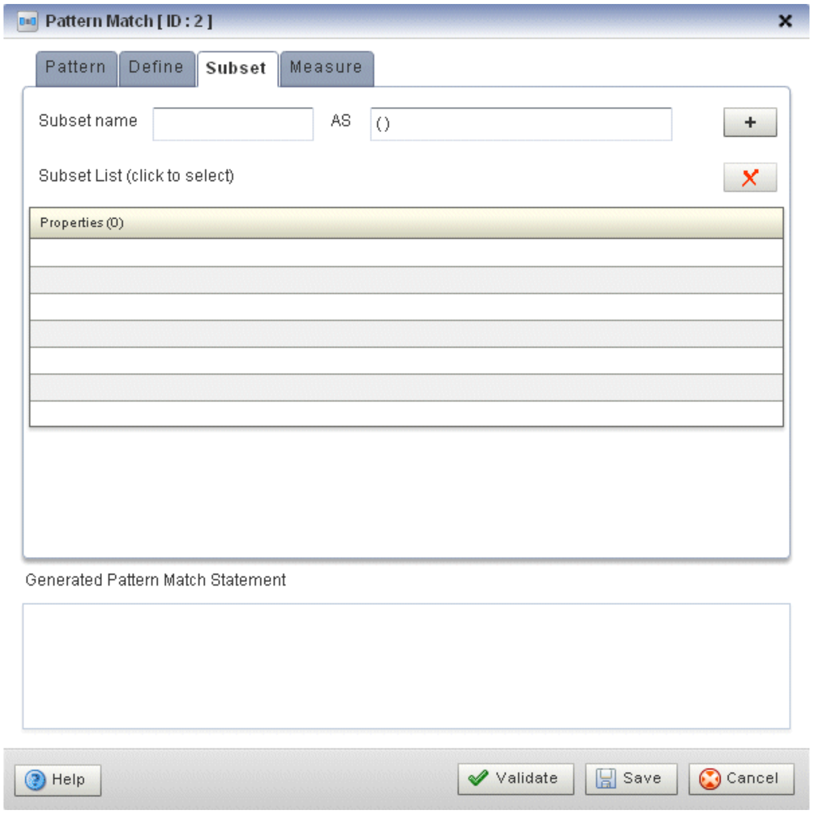
`AS`	Define an expression by clicking the **Expression Builder** button.
	For example:
	`A AS  (A.temp >= 25)`
	`B  AS  ((B.temp >= 25) and`
	`          (B.element_time - A.element_time < INTERVAL "0 00:00:05:00"`
	`DAY TO SECOND))`
	`C  AS  (C.element_time - A.element_time >= INTERVAL "0 00:00:05:00" DAY TO`
	`SECOND)`
	For more information, see Configure an Expression Using the Expression Builder.

`Object List`	A list of the definitions you specify. To add a definition to the **Object List**, click the **Add** button. To remove a definition from the **Object List**, click the **Delete** button.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

5. Click the **Subset** tab.

   The Subset tab appears as Figure 6-45 shows.

Figure 6-45 Query Wizard Pattern Editor Dialog: Subset Tab



6. Edit the Subset tab as Table 6-7 describes.

Table 6-7 Query Wizard Pattern Editor Attributes: Subset Tab

Attribute	Description
`Subset name`	Enter a name for the subset.

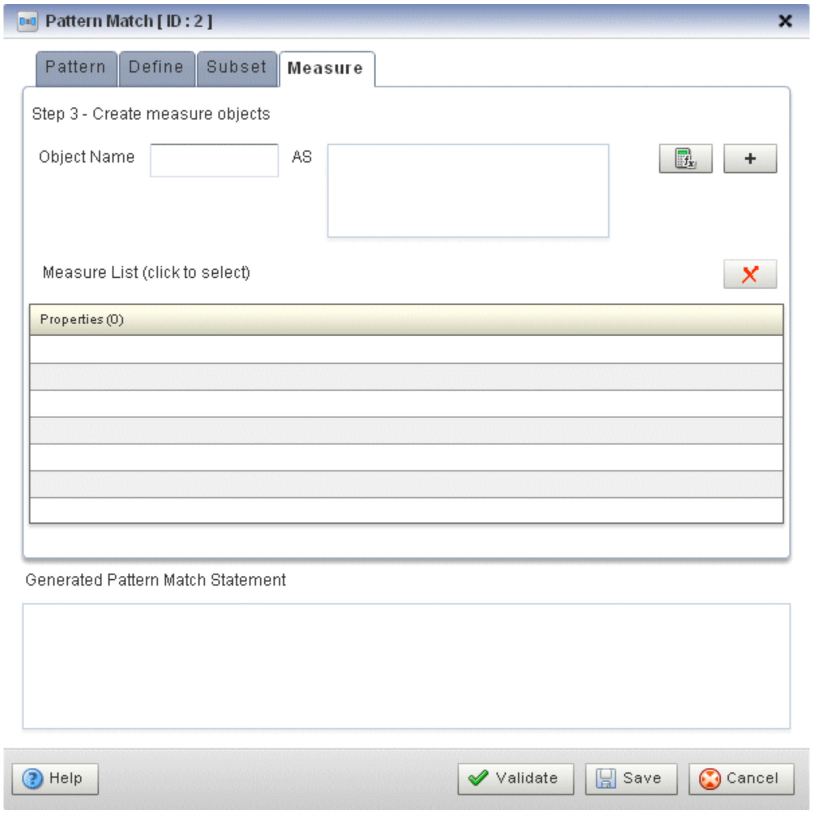
`AS`	Enter a list of the object names as you specified in the **Pattern Expression** attribute of the Pattern tab (see Table 6-4). For example, given the following pattern:
	`(A W+ X+ Y+ Z+)`
	You could define subsets such as:
	`S1 AS (Z)`
	`S2 AS (A)`
	`S3 AS (A,W,Y)`

`Subset List`	A list of the subsets you specify. To add a subset to the **Subset List**, click the **Add** button. To remove a subset from the **Subset List**, click the **Delete** button.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

7. Click the **Measure** tab.

The Measure tab appears as Figure 6-46 shows.

Figure 6-46 Query Wizard Pattern Editor Dialog: Measure Tab



8. Edit the Measure tab as Table 6-8 describes.

Table 6-8 Query Wizard Pattern Editor Attributes: Measure Tab

Attribute	Description

`Object Name`	Enter the name for a measure. For example, if the pattern expression is:
	`(A B* C)`
	Then you could create a measure for any of `A`, `B`, and `C`.
	If you define subsets:
	`S1 AS (Z)`
	`S2 AS (A)`
	`S3 AS (A,W,Y)`
	Then you could create a measure for any of `S1`, `S2`, and `S3`.

Attribute	Description
AS	Define an expression by clicking the **Expression Builder** button. If the pattern expression is: (A B* C) Then you can create a measure for any of A, B, and C. If you define subsets: S1 AS (Z) S2 AS (A) S3 AS (A,W,Y) Then you could create a measure for any of S1, S2, and S3. For example: `sumDecrArm as sum(S3.c2)` For more information, see Configure an Expression Using the Expression Builder.
Measure List	A list of the measures you specify. To add a measure to the **Properties** list, click the **Add** button. To remove a measure from the **Properties** list, click the **Delete** button.
Generated CQL Statement	Read-only Oracle CQL fragment associated with this CQL Construct.

9. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

10. Click **Save**.

    The Pattern icon displays a green check mark to indicate that it is configured.

6.1.9.5 Configure a Select CQL Construct

The Select Oracle CQL construct represents the Oracle CQL `SELECT` clause. For information on representing the `WHERE` clause, see Configure a Filter CQL Construct.

The Select Oracle CQL construct editor provides a tab for each of the following sub-clauses:

- `select_clause`: Use this clause to specify the stream elements to select from the stream or view you specify.

  See Figure 6-48.

- `opt_group_by_clause`: Use this clause to specify optional grouping conditions your query applies to its results.

  See Figure 6-49.

- `opt_having_clause`: Use this clause to restrict the groups of returned stream elements to those groups for which the specified *`condition`* is `TRUE`. If you omit this clause, then Oracle Event Processing returns summary results for all groups.

  See Figure 6-50.

- `order_by_clause`: Use this clause to specify optional ordering conditions your query applies to its results.

  See Figure 6-51.

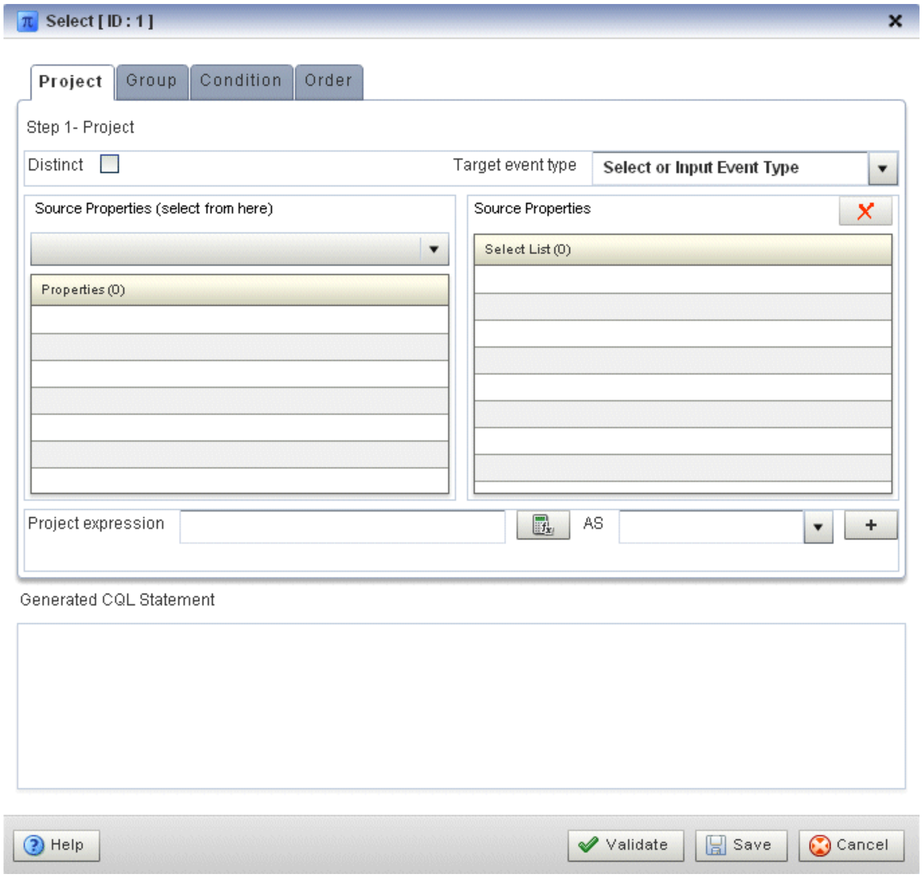
Configure a Select CQL Construct

1. Double click the **Select** icon as Figure 6-47 shows.

   Figure 6-47 Select Icon

   The Select editor dialog appears as Figure 6-48 shows.

   Figure 6-48 Query Wizard Select Editor Dialog: Project Tab

2. Edit the Project tab as Table 6-9 describes.

   The general procedure is:

   a. Select a **Source**.

   b. Select a **Target event type**.

   c. Double click a property in the Properties list to add the property to the **Project expression** field.

   d. Click the **Expression Builder** button and create a **Project expression**.

e. In the **AS** field, enter an alias for the project expression result or select one of the **Target event type** attributes.

f. Click the **Add** button.

   To remove a project expression, select it in the Project predicates list and click **Delete**.

g. Optionally, check **Distinct**.

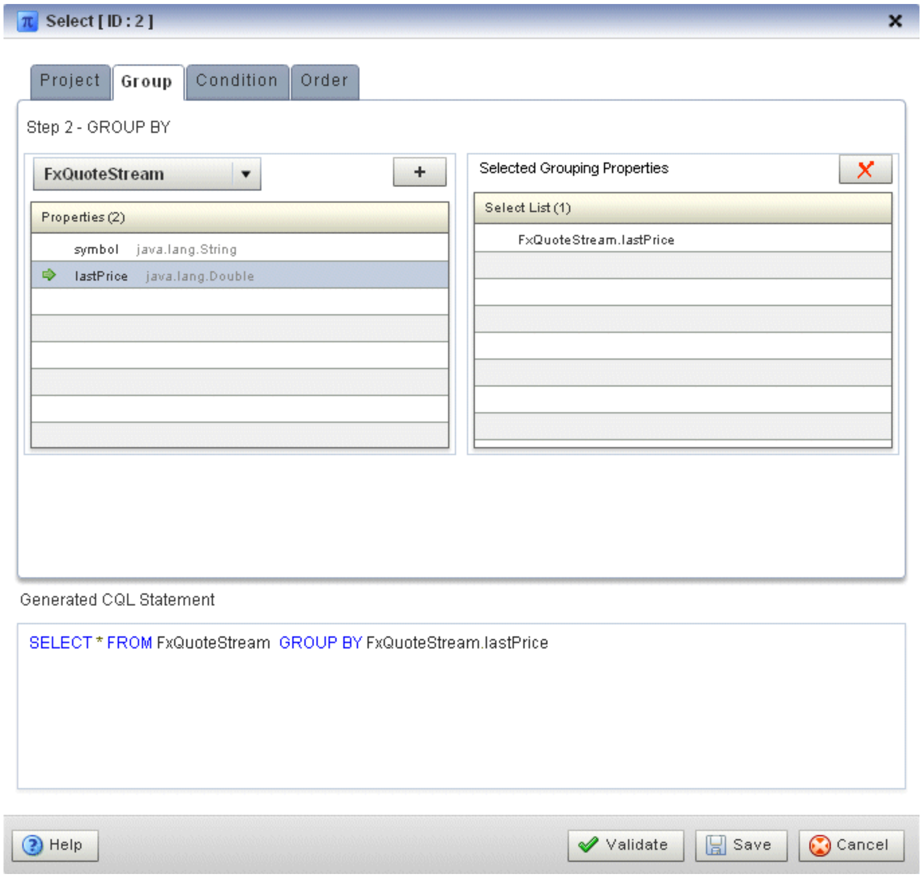
Table 6-9 Query Wizard Select Editor Attributes: Project Tab

Attribute	Description
`Distinct`	Check this option if you want Oracle Event Processing to return only one copy of each set of duplicate tuples selected. Uncheck this option if you want Oracle Event Processing to return all tuples selected, including duplicates. Duplicate tuples are those with matching values for each expression in the select list. Oracle Event Processing does not support nested aggregations.
`Target event type`	Select or input the name of the event expected by the down-stream channel you connected to this Oracle CQL processor.
`Source`	Select the name of the stream, relation, or view Source CQL Construct that you connected to this Project CQL construct. For more information, see Configure an RSource CQL Construct.
`Properties`	Read-only list of the properties of the event offered by this source. For more information, see Configure an RSource CQL Construct.
`Project Predicates`	A list of the project expressions you define. To add a project expression to the **Project Predicates** list, click the **Add** button. To remove a project expression from the **Project Predicates** list, select the project expression in the Project Predicates list and click the **Delete** button.
`Project Expressions`	Define an expression by clicking the **Expression Builder** button. For example: `LENGTH( FxQuoteStream.symbol < 100 )` For more information, see Configure an Expression Using the Expression Builder.
`AS`	Enter an alias for the project expression result or select one of the **Target event type** attributes.
`Generated CQL Statement`	Read-only Oracle CQL fragment associated with this CQL Construct.

3. Click the **Group** tab.

   The Group tab editor dialog appears as Figure 6-49 shows.

Figure 6-49 Query Wizard Select Editor Dialog: Group Tab



Properties (2)	
symbol	java.lang.String
lastPrice	java.lang.Double

Select List (1)
FxQuoteStream.lastPrice

Generated CQL Statement

```
SELECT * FROM FxQuoteStream  GROUP BY FxQuoteStream.lastPrice
```

4. Edit the Group tab as Table 6-10 describes.

   The general procedure is:

   a. Select a source from the pull down menu.

   b. Select a property in the **Properties** list.

   c. To add the selected property to the **Grouping properties** list, click the **Add** button.

      To remove a property from the Grouping properties list, select the property in the Grouping properties list and click the **Delete** button.

Table 6-10 Query Wizard Select Editor Attributes: Group Tab

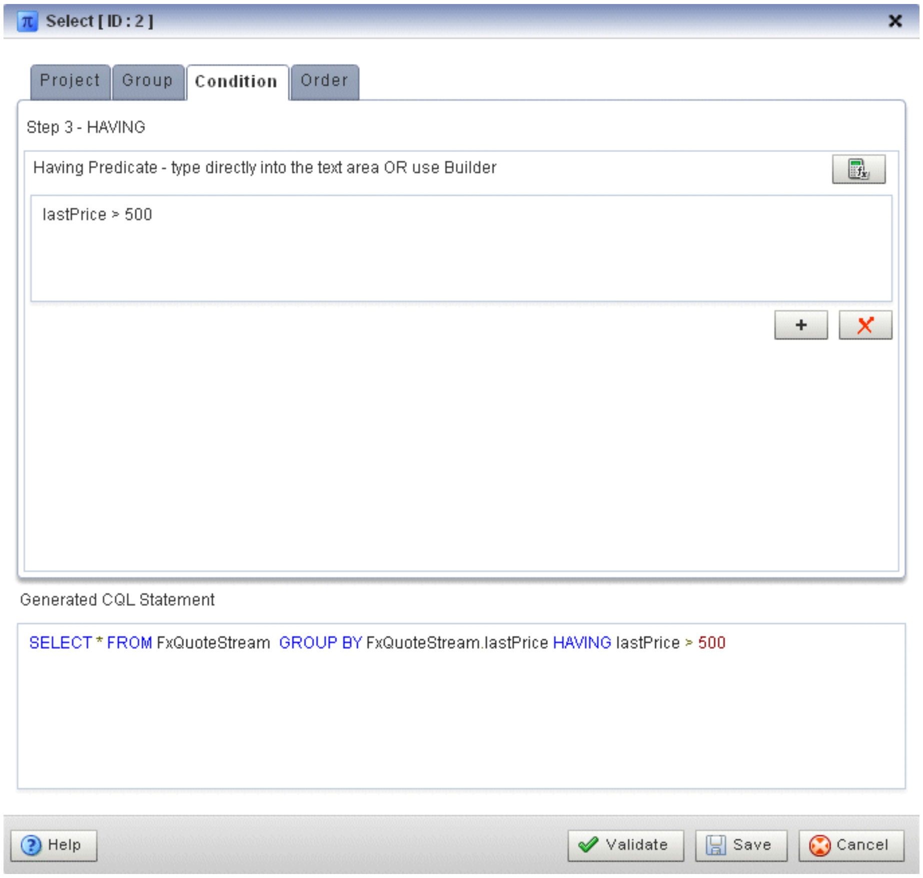
Attribute	Description
`Select a source`	Select a source from the pull-down menu.
`Properties`	The properties of the event offered by the selected source.

Attribute	Description
`Grouping Properties`	A list of the grouping properties you define. To add the selected property to the Grouping properties list, click the **Add** button. To remove a property from the Grouping properties list, select the property in the Grouping properties list and click the **Delete** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

5. Click the **Condition** tab.

   The Condition tab editor dialog appears as Figure 6-50 shows.

Figure 6-50 Query Wizard Select Editor Dialog: Condition Tab



Step 3 - HAVING

Having Predicate - type directly into the text area OR use Builder

lastPrice > 500

Generated CQL Statement

```
SELECT * FROM FxQuoteStream  GROUP BY FxQuoteStream.lastPrice HAVING lastPrice > 500
```

6. Edit the Condition tab as Table 6-11 describes.

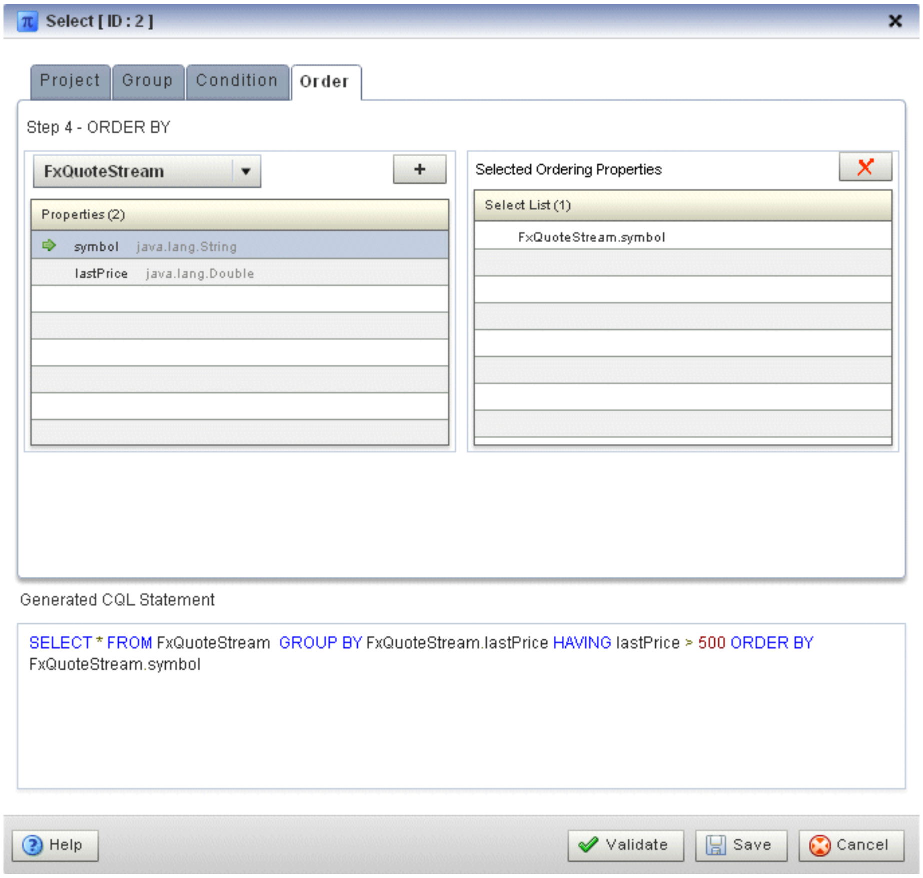
Table 6-11 Query Wizard Select Editor Attributes: Condition Tab

Attribute	Description
Having Predicate	Define an expression by entering a having predicate directly or clicking the **Expression Builder** button. For example: `lastPrice > 500` For more information, see Configure an Expression Using the Expression Builder. To add a condition expression to the generated CQL statement, click the **Add** button. To remove the condition expression from the generated CQL statement, click the **Delete** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

7. Click the **Order** tab.

The Order tab editor dialog appears as Figure 6-51 shows.

Figure 6-51 Query Wizard Select Editor Dialog: Order Tab



```
SELECT * FROM FxQuoteStream  GROUP BY FxQuoteStream.lastPrice HAVING lastPrice > 500 ORDER BY
FxQuoteStream.symbol
```

8. Edit the Order tab as Table 6-12 describes.

The general procedure is:

a. Select a source from the pull down menu.

b. Select a property in the **Properties** list.

c. To add the selected property to the **Ordering properties** list, click the **Add** button.

   To remove a property from the Ordering properties list, select the property in the Ordering properties list and click the **Delete** button.

Table 6-12 Query Wizard Select Editor Attributes: Order Tab

Attribute	Description
`Select a source`	Select a source from the pull-down menu.
`Properties`	The properties of the event offered by the selected source.
`Ordering Properties`	A list of the ordering properties you define. To add the selected property to the Ordering properties list, click the **Add** button. To remove a property from the Ordering properties list, select the property in the Ordering properties list and click the **Delete** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

9. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

10. Click **Save**.

    The Select icon displays a green check mark to indicate that it is configured.

6.1.9.6 Configure a Join CQL Construct Using + Syntax

The Join Oracle CQL construct represents an Oracle CQL join between two or more sources. You can create and inner joins and left and right outer joins using the (deprecated) + syntax.

Note:

If your Oracle CQL rule uses the `LEFT` or `RIGHT OUTER JOIN` clause, see Configure a Join CQL Construct Using LEFT or RIGHT OUTER JOIN.

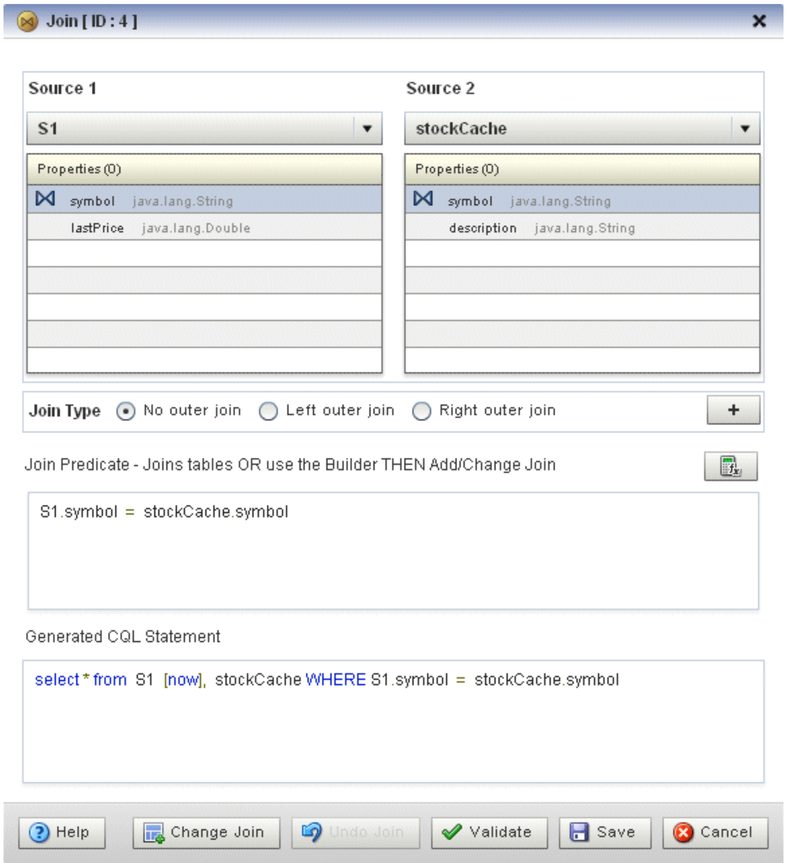
Configure a Join CQL Construct

1. Double click the **Join** icon as Figure 6-52 shows.

Figure 6-52 Join Icon

The Join editor dialog appears as Figure 6-53 shows.

Figure 6-53 Query Wizard Join Editor Dialog



Join [ID : 4]

Source 1: S1

Properties (0)	
symbol	java.lang.String
lastPrice	java.lang.Double

Source 2: stockCache

Properties (0)	
symbol	java.lang.String
description	java.lang.String

Join Type ● No outer join ○ Left outer join ○ Right outer join

Join Predicate - Joins tables OR use the Builder THEN Add/Change Join

```
S1.symbol = stockCache.symbol
```

Generated CQL Statement

```
select * from S1 [now], stockCache WHERE S1.symbol = stockCache.symbol
```

2. Edit the dialog as Table 6-13 describes.

Table 6-13 Query Wizard Join Editor Attributes

Attribute	Description
`Source 1`	Select the first source to join.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to join.
`Properties`	Read-only list of properties of Source 2.
`Join Type`	Select the type of join: • **No outer join**: select this option to create a simple join. See *Oracle Fusion Middleware Oracle CQL Language Reference for Oracle Event Processing*. • **Left outer join**: select this option to create a left outer join. See *Oracle Fusion Middleware Oracle CQL Language Reference for Oracle Event Processing*. • **Right outer join**: select this option to create a right outer join. See *Oracle Fusion Middleware Oracle CQL Language Reference for Oracle Event Processing*.

Attribute	Description
`Join Predicate`	Create the **Join Predicate** by doing either of the following: • Click the **Plus Sign** button to create the join predicate using the Source 1 and Source 2 properties you select. • Click the **Expression Builder** button to create the join predicate using any of the Source 1 and Source 2 properties and Oracle CQL functions and operators.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Select a source from the **Source 1** pull-down menu.
4. Select a source from the **Source 2** pull-down menu.
5. Select a property in the Source 1 **Properties** list to join on.
6. Select a property in the Source 2 **Properties** list to join on.
7. Choose the **Join Type**.
8. Create the **Join Predicate** by doing either of the following:
   a. Click the **Plus Sign** button to create the join predicate using the Source 1 and Source 2 properties you select.
   b. Click the **Expression Builder** button to create the join predicate using any of the Source 1 and Source 2 properties and Oracle CQL functions and operators.
9. To add the join predicate to the generated CQL statement, click the **Add Join** button.
10. To remove the join predicate from the generated CQL statement, click the **Undo Join** button.
11. Click the **Validate** button to ensure that your changes are syntactically correct.

    Correct any errors before proceeding.
12. Click **Save**.

    The Join icon displays a green check mark to indicate that it is configured.

6.1.9.7 Configure a Join CQL Construct Using LEFT or RIGHT OUTER JOIN

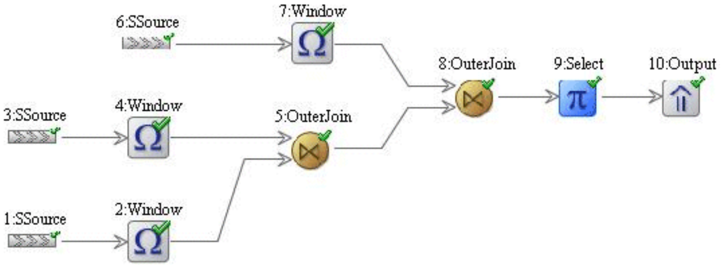
The Join Oracle CQL construct represents an Oracle CQL join between two or more sources. You can view the configuration of an Oracle CQL rule using the `LEFT` or `RIGHT OUTER` join syntax.

Note:

If your Oracle CQL rule uses an inner join or outer join using the deprecated + syntax, see Configure a Join CQL Construct Using + Syntax.

Figure 6-54 shows an example Oracle CQL outer join in the query constructor.

Figure 6-54 Oracle CQL Outer Join



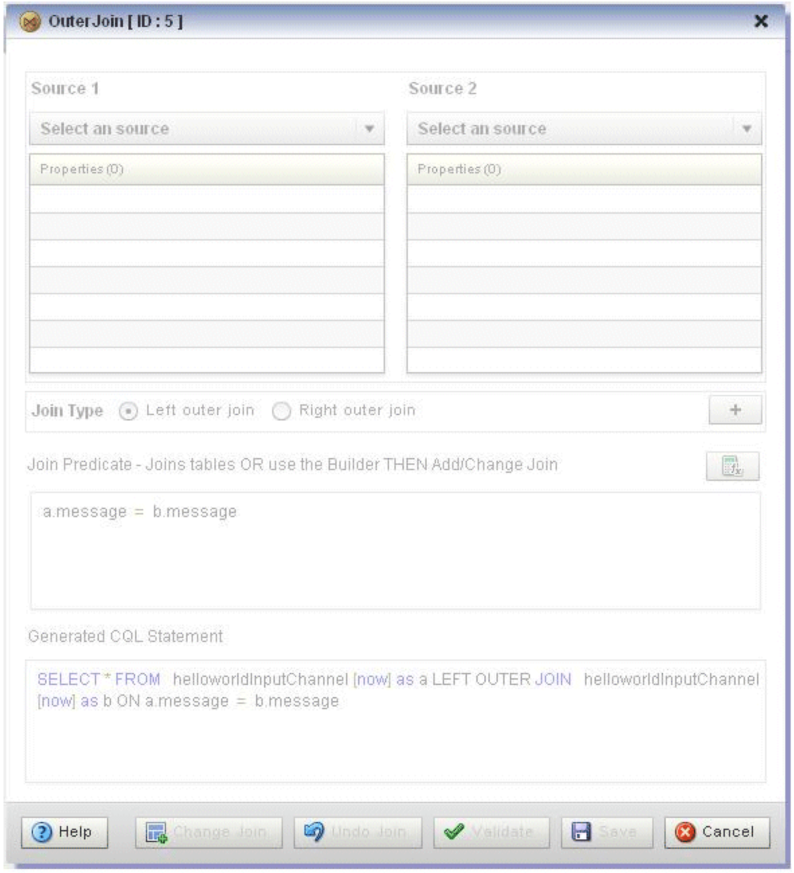
Configure a Join CQL Construct Using LEFT or RIGHT OUTER JOIN:

1. Double click the **Join** icon as Figure 6-55 shows.

   Figure 6-55 Join Icon

   The Join editor dialog appears as Figure 6-56 shows.

   Figure 6-56 Query Wizard Join Editor Dialog: LEFT or RIGHT OUTER JOIN Syntax

   OuterJoin [ID : 5]

   Source 1: Select an source

   Properties (0)

   Source 2: Select an source

   Properties (0)

   Join Type ● Left outer join ○ Right outer join

   Join Predicate - Joins tables OR use the Builder THEN Add/Change Join

   ```
   a.message = b.message
   ```

   Generated CQL Statement

   ```
   SELECT * FROM  helloworldInputChannel [now] as a LEFT OUTER JOIN  helloworldInputChannel
   [now] as b ON a.message = b.message
   ```

6.1.9.8 Configure a Window CQL Construct

The Window Oracle CQL construct represents an Oracle CQL stream-to-relation operator.

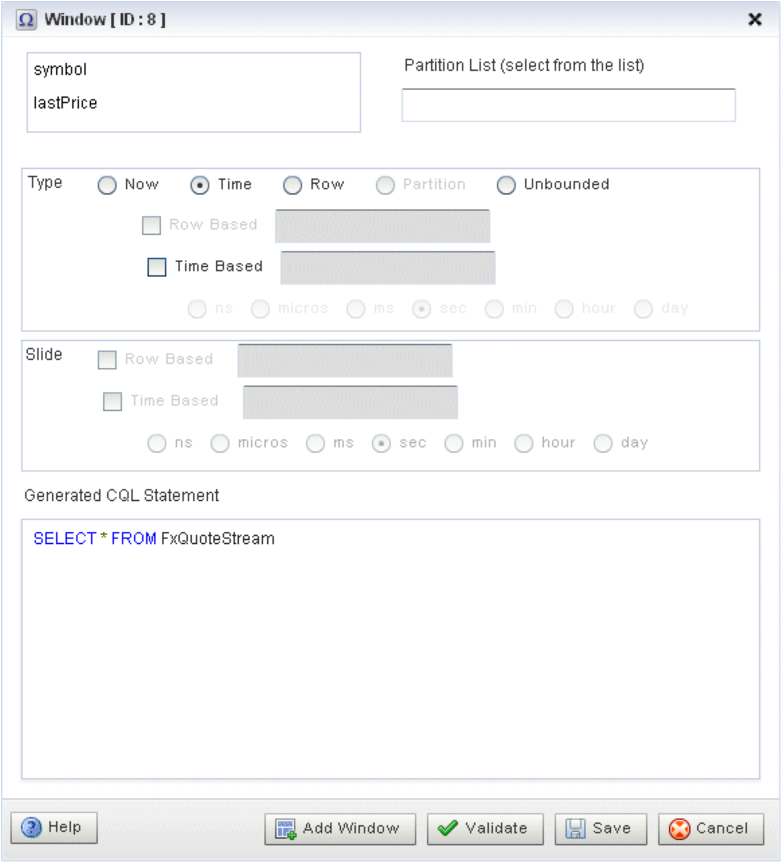
Configure a Window CQL Construct

1. Double click the **Window** icon as Figure 6-57 shows.

   Figure 6-57 Window Icon

   The Window editor dialog appears as Figure 6-58 shows.

   Figure 6-58 Query Wizard Window Editor Dialog

   symbol
   lastPrice

   Partition List (select from the list)

   Type ○ Now ● Time ○ Row ○ Partition ○ Unbounded
   □ Row Based
   □ Time Based
   ○ ns ○ micros ○ ms ● sec ○ min ○ hour ○ day

   Slide □ Row Based
   □ Time Based
   ○ ns ○ micros ○ ms ● sec ○ min ○ hour ○ day

   Generated CQL Statement

   ```
   SELECT * FROM FxQuoteStream
   ```

2. Edit the dialog as Table 6-14 describes.

Table 6-14 Query Wizard Window Editor Attributes

Attribute	Description
`Partition` List	Select one or more stream elements to add to the partition list if you want to create a partitioned window. To remove stream elements from the Partition List, click and drag to select the values in the Partition List and press the **Delete** key on your keyboard.
`Type`	The type of window to create: **Now**: Select this option to create a time-based range window that defines its output relation such that, when `T = 0`, the relation at time `t` consists of tuples obtained from elements of `S` with timestamp `t`. **Time**: Select this option to create a time-based range window that defines its output relation over time by sliding an interval of size `T` time units capturing the latest portion of an ordered stream. Elements that have been in the window for more than `T` time units are subject to deletion. **Row**: Select this option to create a tuple-based window that defines its output relation over time by sliding a window of the last `N` tuples of an ordered stream. For the output relation `R` of `S [rows N]`, the relation at time `t` consists of the `N` tuples of `S` with the largest timestamps `<= t` (or all tuples if the length of `S` up to `t` is `<= N`). Optionally, specify a row-based slide for this window using the **Slide** attribute. **Partition**: Select this option to create any of the following partitioned windows: • "S [partition by A1,..., Ak rows N]" • "S [partition by A1,..., Ak rows N range T]" The Partition type is inactive unless you add one or more stream elements to the **Partition List**. Optionally, specify a time-based slide for this window using the **Slide** attribute. **Unbounded**: Select this option to create a time-based range window defines its output relation such that, when `T = infinity`, the relation at time `t` consists of tuples obtained from all elements of `S` up to `t`. Elements remain in the window indefinitely.
`Slide`	Specify a sliding window: **Row Based**: Select this option to create any of the following row-based sliding windows: • "S[rows N1 slide N2]" **Time Based**: Select this option to create any of the following time-based sliding windows: • "S[range T1 slide T2]" • "S [partition by A1,..., Ak rows N range T1 slide T2]"
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. If you want a partitioned window, select the stream elements you want in the Partition List.

   To remove stream elements from the Partition List, click and drag to select the values in the Partition List and press the **Delete** key on your keyboard.

4. Select the **Type**.

   The **Partition** type is inactive unless you add one or more stream elements to the Partition List.

5. Select **Row Based** or **Time Based** for the selected type.

6. Enter the size of the window:

   a. For a Row Based type, enter a number of rows (tuples or stream elements).

   b. For a Time Based type, enter a number of time units and select the time unit.

7. Select a **Slide**:

   a. For a Row Based type, enter a number of rows (tuples or stream elements).

   b. For a Time Based type, enter a number of time units and select the time unit.

8. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

9. To add the window to the generated CQL statement, click the **Add Window** button.

10. Click **Save**.

    The Window icon displays a green check mark to indicate that it is configured.

6.1.9.9 Configure a Filter CQL Construct

The Filter Oracle CQL construct represents an Oracle CQL `WHERE` clause. For information on representing the `SELECT` and `FROM` clauses, see Configure a Select CQL Construct.

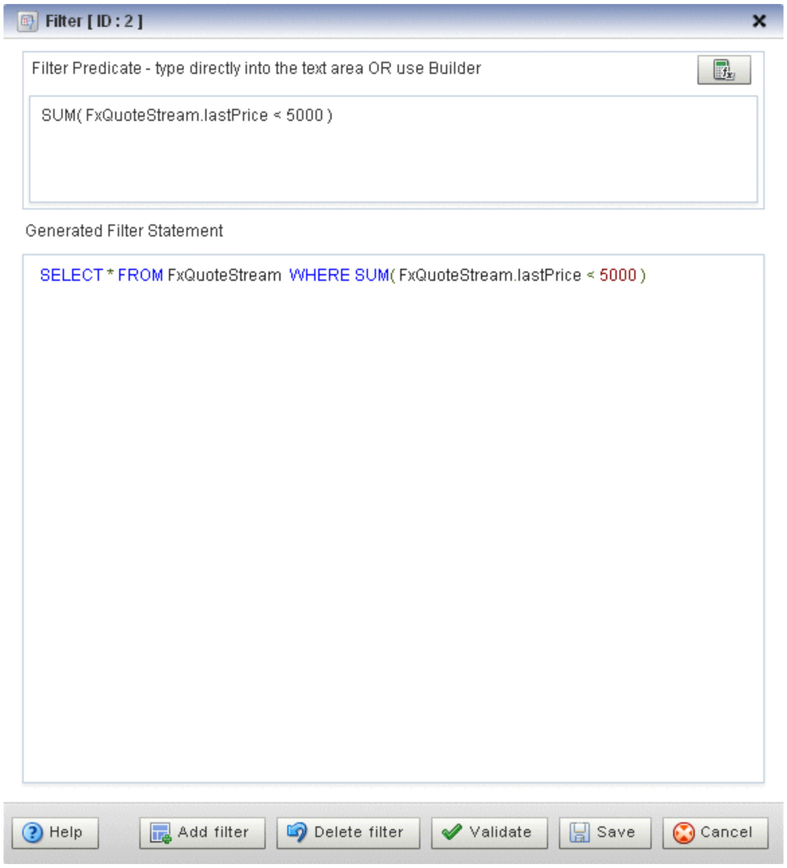
Configure a Filter CQL Construct

1. Double click the **Filter** icon as Figure 6-59 shows.

   Figure 6-59 Filter Icon

   The Filter editor dialog appears as Figure 6-60 shows.

Figure 6-60 Query Wizard Filter Editor Dialog



Filter [ID : 2]
Filter Predicate - type directly into the text area OR use Builder

```
SUM( FxQuoteStream.lastPrice < 5000 )
```

Generated Filter Statement

```
SELECT * FROM FxQuoteStream  WHERE SUM( FxQuoteStream.lastPrice < 5000 )
```

Help
Add filter
Delete filter
Validate
Save
Cancel

2. Edit the dialog as Table 6-15 describes.

Table 6-15 Query Wizard Filter Editor Attributes

Attribute	Description
`Filter Predicate`	Define an expression by entering a filter predicate directly or clicking the **Expression Builder** button. For example: `SUM( FxQuoteStream.lastPrice < 5000 )` For more information, see Configure an Expression Using the Expression Builder. To add the filter predicate to the generated CQL statement, click the **Add Filter** button. To remove the filter predicate from the generated CQL statement, click the **Delete Filter** button.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Define the filter predicate by doing one of the following:

   a. Enter the filter predicate directly.

   b. Click the **Expression Builder** button.

For more information, see Configure an Expression Using the Expression Builder.

4. To add a condition expression to the generated CQL statement, click the **Add Filter** button.

5. To remove the condition expression from the generated CQL statement, click the **Delete Filter** button.

6. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

7. Click **Save**.

   The Source icon displays a green check mark to indicate that it is configured.

6.1.9.10 Configure a Union CQL Construct

The Union Oracle CQL construct represents an Oracle CQL `UNION` or `UNION ALL` clause. You can perform a union between two relations; relation can be emitted either from a view or a channel. You can perform a `union all` between two relations or two streams. You cannot perform a union between a relation and a stream.

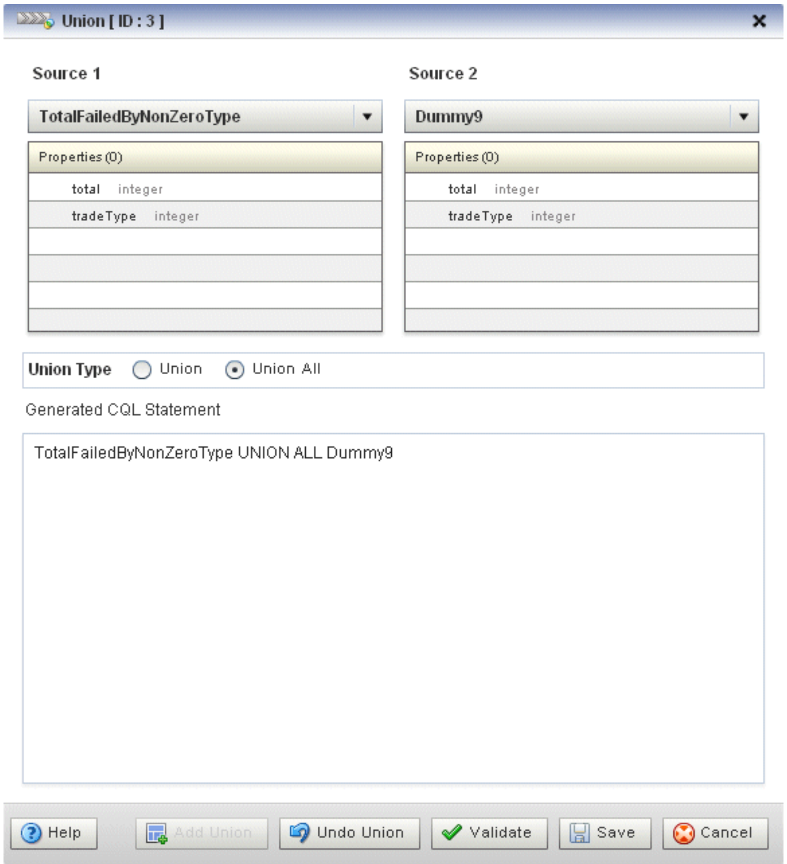
Configure a Union CQL Construct

1. Double click the **Union** icon as Figure 6-61 shows.

   Figure 6-61 Union Icon

   The Union editor dialog appears as Figure 6-62 shows.

Figure 6-62 Query Wizard Union Editor Dialog



TotalFailedByNonZeroType UNION ALL Dummy9

2. Edit the dialog as Table 6-16 describes.

Table 6-16 Query Wizard Union Editor Attributes

Attribute	Description
`Source 1`	Select the first source to union.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to union.
`Properties`	Read-only list of properties of Source 2.
`Union Type`	Select the type of union: • **Union**: select this to create a `UNION` statement. • **Union All**: select this option to create a `UNION ALL` statement.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. Select a source from the **Source 1** pull-down menu.

4. Select a source from the **Source 2** pull-down menu.

5. Choose the **Union Type**.

6. To add the union clause to the generated CQL statement, click the **Add Union** button.

7. To remove the union clause from the generated CQL statement, click the **Undo Union** button.

8. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

9. Click **Save**.

   The Union icon displays a green check mark to indicate that it is configured.

6.1.9.11 Configure an Intersect CQL Construct

The Intersect Oracle CQL construct represents an Oracle CQL `INTERSECT` clause. You can perform an intersect between two relations. You cannot perform an intersect between a relation and a stream or between two streams.

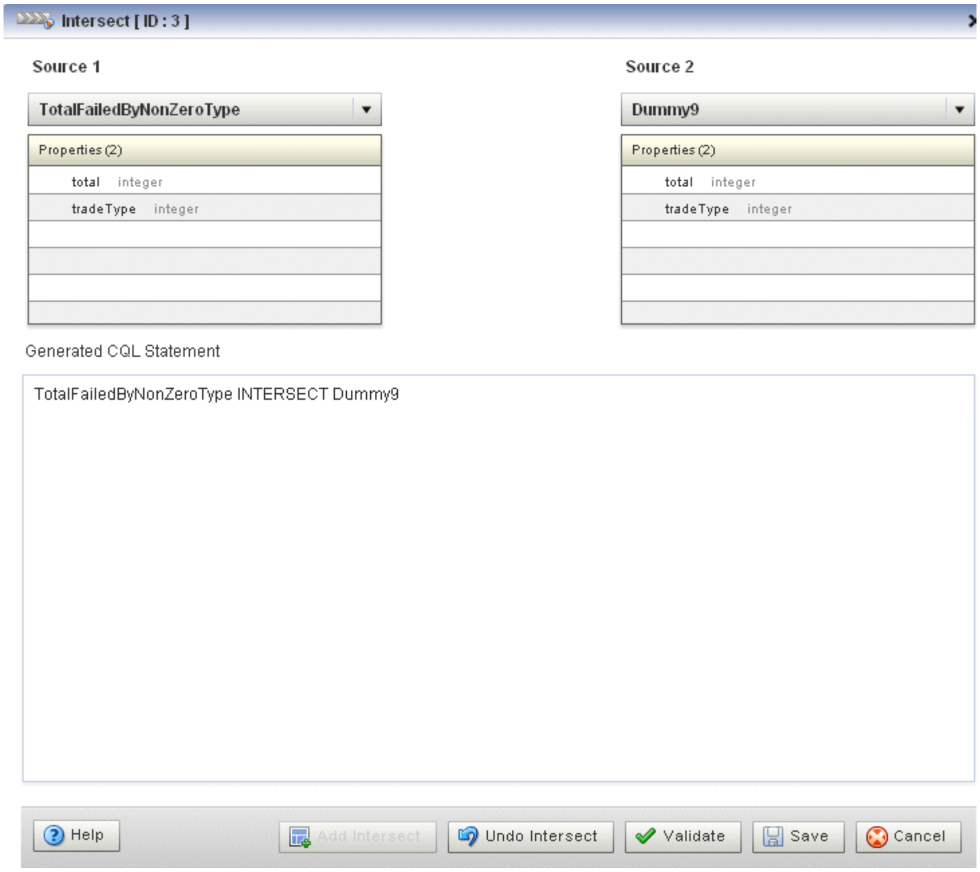
Configure an Intersect CQL Construct

1. Double click the **Intersect** icon as Figure 6-63 shows.

   Figure 6-63 Intersect Icon

   The Intersect editor dialog appears as Figure 6-64 shows.

Figure 6-64 Query Wizard Intersect Editor Dialog



TotalFailedByNonZeroType INTERSECT Dummy9

2. Edit the dialog as Table 6-17 describes.

Table 6-17 Query Wizard Intersect Editor Attributes

Attribute	Description
`Source 1`	Select the first source to intersect.
`Properties`	Read-only list of properties of Source 1.
`Source 2`	Select the second source to intersect.
`Properties`	Read-only list of properties of Source 2.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To add the intersect clause to the generated CQL statement, click the **Add Intersect** button.

4. To remove the intersect clause from the generated CQL statement, click the **Undo Intersect** button.

5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

6. Click **Save**.

The Intersect icon displays a green check mark to indicate that it is configured.

6.1.9.12 Configure a Minus CQL Construct

The Minus Oracle CQL construct represents an Oracle CQL `MINUS` clause. You can perform a minus between two relations. You cannot perform a minus between a relation and a stream or between two streams.

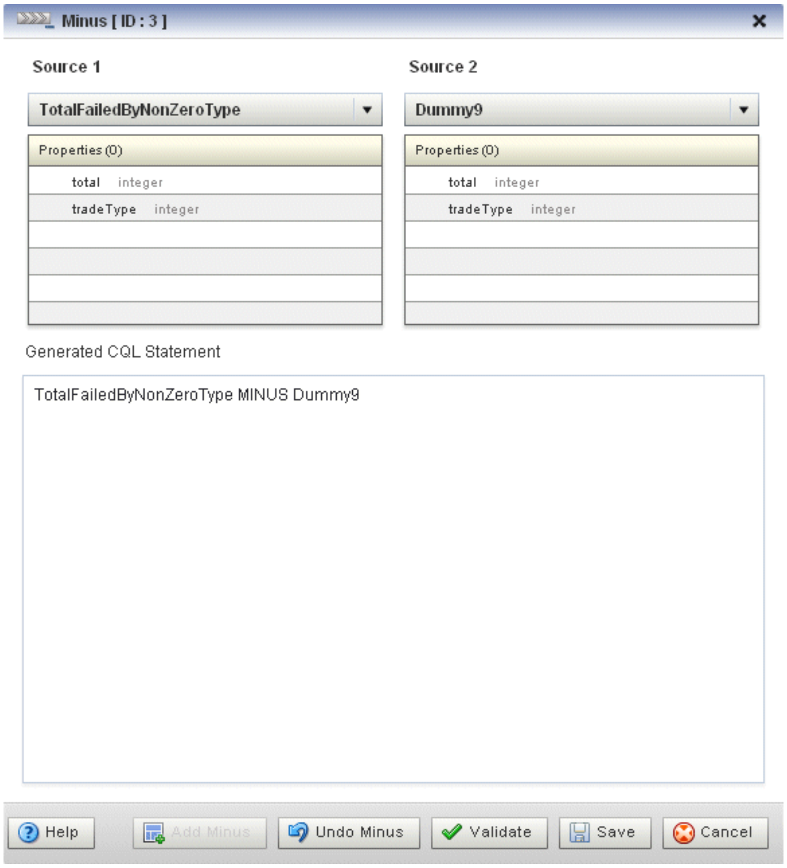
Configure a Minus CQL Construct

1. Double click the **Intersect** icon as Figure 6-65 shows.

   Figure 6-65 Minus Icon

   The Minus editor dialog appears as Figure 6-66 shows.

   Figure 6-66 Query Wizard Minus Editor Dialog

   Minus [ID : 3]

   Source 1: TotalFailedByNonZeroType

   Properties (0)

total	integer
tradeType	integer

   Source 2: Dummy9

   Properties (0)

total	integer
tradeType	integer

   Generated CQL Statement

   ```
   TotalFailedByNonZeroType MINUS Dummy9
   ```

2. Edit the dialog as Table 6-18 describes.

   Table 6-18 Query Wizard Minus Editor Attributes

Attribute	Description
`Source 1`	Select the first source to minus.
`Properties`	Read-only list of properties of Source 1.

Attribute	Description
`Source 2`	Select the second source to minus.
`Properties`	Read-only list of properties of Source 2.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To add the minus clause to the generated CQL statement, click the **Add Minus** button.

4. To remove the minus clause from the generated CQL statement, click the **Undo Minus** button.

5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

6. Click **Save**.

   The Minus icon displays a green check mark to indicate that it is configured.

6.1.9.13 Configure an IStream CQL Construct

The IStream Oracle CQL construct represents an Oracle CQL IStream relation-to-stream operator. `Istream` (for "Insert stream") applied to a relation `R` contains `(s,t)` whenever tuple `s` is in `R(t) - R(t-1)`, that is, whenever `s` is inserted into `R` at time `t`. If a tuple happens to be both inserted and deleted with the same timestamp then `IStream` does not output the insertion.

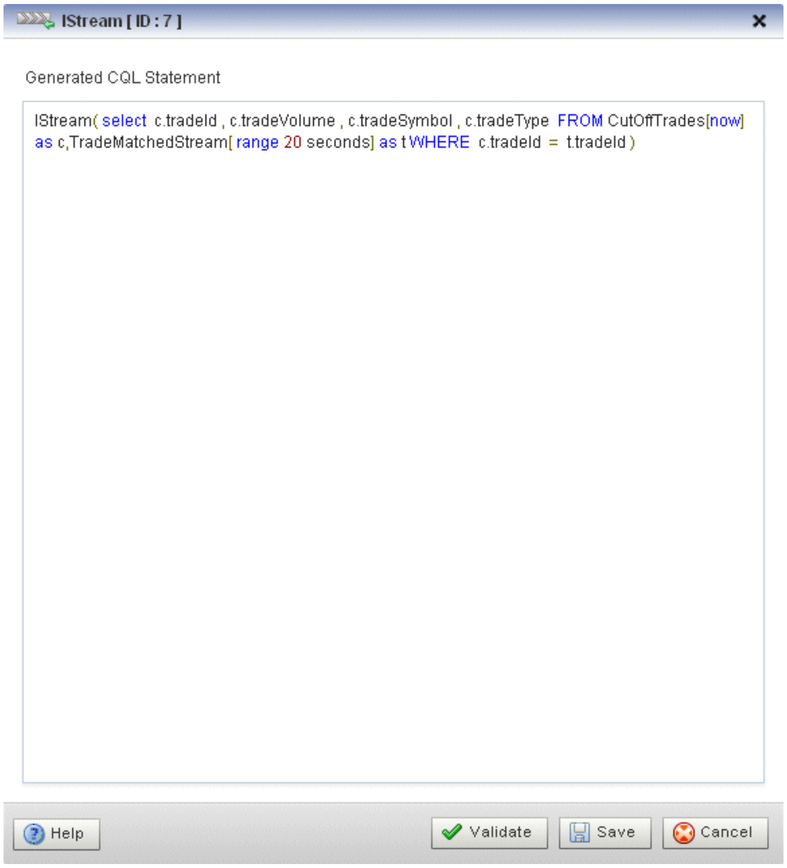
Configure an IStream CQL Construct

1. Double click the **IStream** icon as Figure 6-67 shows.

   Figure 6-67 IStream Icon

   The IStream editor dialog appears as Figure 6-68 shows.

Figure 6-68 Query Wizard IStream Editor Dialog



```
IStream( select  c.tradeId , c.tradeVolume , c.tradeSymbol , c.tradeType  FROM CutOffTrades[now]
as c,TradeMatchedStream[ range 20 seconds] as t WHERE  c.tradeId  =  t.tradeId )
```

2. Edit the dialog as Table 6-19 describes.

Table 6-19 Query Wizard IStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `IStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The IStream icon displays a green check mark to indicate that it is configured.

6.1.9.14 Configure a DStream CQL Construct

The DStream Oracle CQL construct represents an Oracle CQL DStream relation-to-stream operator. `Dstream` (for "Delete stream") applied to a relation `R` contains `(s,t)` whenever tuple `s` is in `R(t-1) - R(t)`, that is, whenever `s` is deleted from `R` at time `t`.

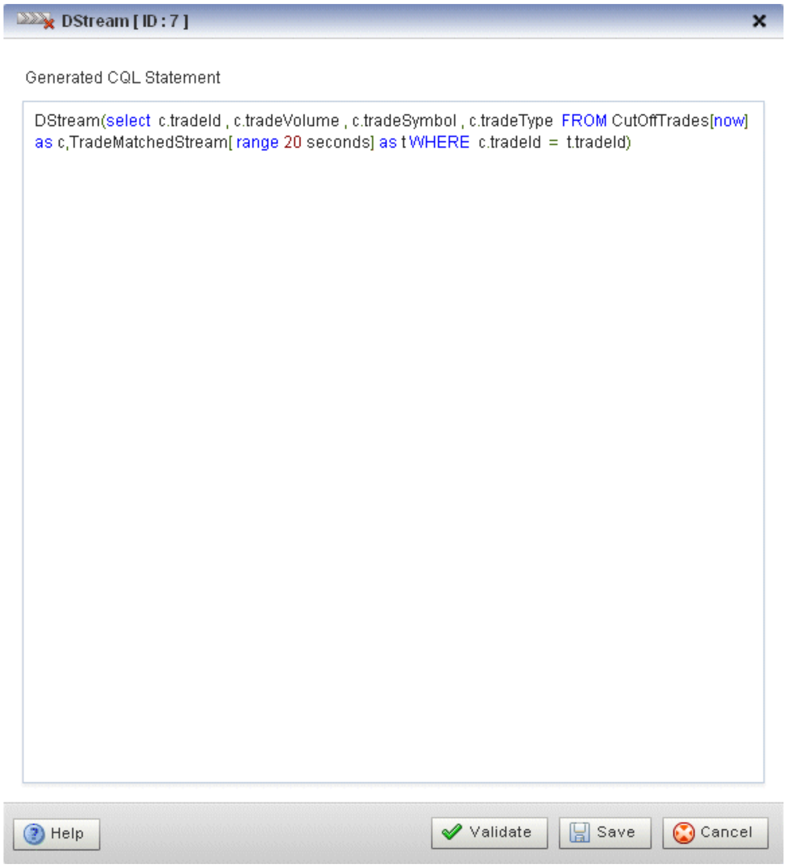
Configure a DStream CQL Construct

1. Double click the **DStream** icon as Figure 6-69 shows.

   Figure 6-69 DStream Icon

   The DStream editor dialog appears as Figure 6-70 shows.

   Figure 6-70 Query Wizard DStream Editor Dialog

   Generated CQL Statement

   ```
   DStream(select c.tradeId , c.tradeVolume , c.tradeSymbol , c.tradeType FROM CutOffTrades[now]
   as c,TradeMatchedStream[ range 20 seconds] as t WHERE c.tradeId = t.tradeId)
   ```

2. Edit the dialog as Table 6-20 describes.

   Table 6-20 Query Wizard DStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `DStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The Dstream icon displays a green check mark to indicate that it is configured.

6.1.9.15 Configure an RStream CQL Construct

The RStream Oracle CQL construct represents an Oracle CQL RStream relation-to-stream operator. The `RStream` operator maintains the entire current state of its input relation and outputs all of the tuples as insertions at each time step. Since `Rstream` outputs the entire state of the relation at every instant of time, it can be expensive if the relation set is not very small.

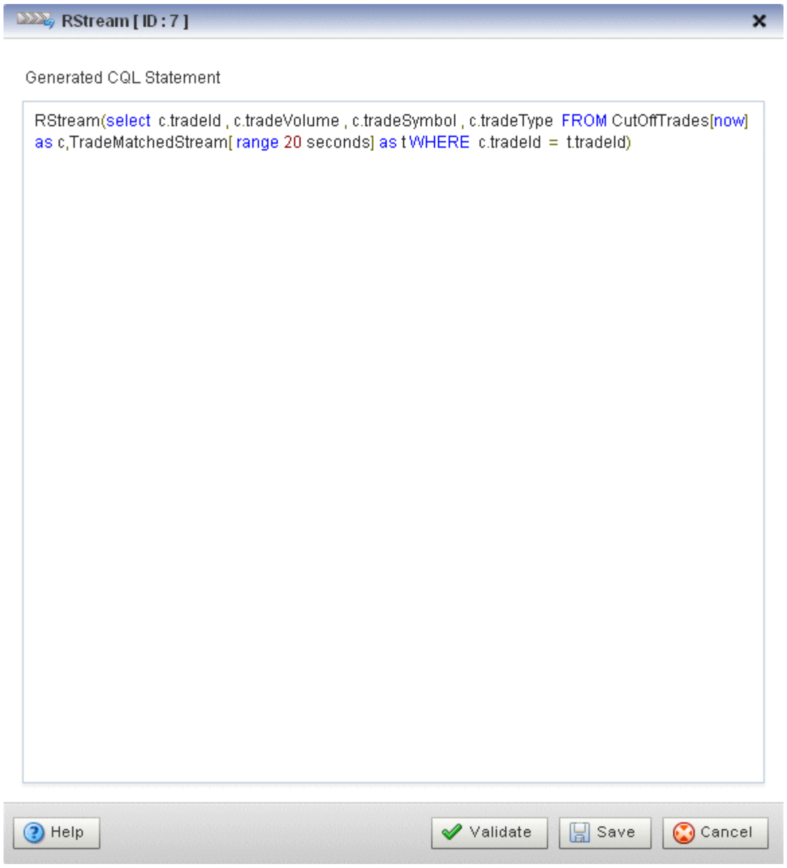
Configure an RStream CQL Construct

1. Double click the **RStream** icon as Figure 6-71 shows.

   Figure 6-71 RStream Icon

   The RStream editor dialog appears as Figure 6-72 shows.

   Figure 6-72 Query Wizard RStream Editor Dialog

   Generated CQL Statement

   ```
   RStream(select c.tradeId , c.tradeVolume , c.tradeSymbol , c.tradeType FROM CutOffTrades[now]
   as c,TradeMatchedStream[ range 20 seconds] as t WHERE c.tradeId = t.tradeId)
   ```

2. Edit the dialog as Table 6-21 describes.

Table 6-21 Query Wizard RStream Editor Attributes

Attribute	Description
`Generated CQL Statement`	Wrap the editable Oracle CQL fragment associated with this CQL Construct in: `RStream()`

3. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

4. Click **Save**.

   The RStream icon displays a green check mark to indicate that it is configured.

6.1.9.16 Configure an Output CQL Construct

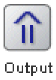
The Output Oracle CQL construct represents the complete Oracle CQL query or view. Using the Output CQL construct, you can:

- Review and edit the complete Oracle CQL statement.
- Create a copy of the Oracle CQL statement and inject it into the Oracle CQL processor under a different name. This is a convenient way of duplicating and modifying an existing statement when you need to create multiple statements that share a common structure.
- Convert between a view and query.

Configure an Output CQL Construct

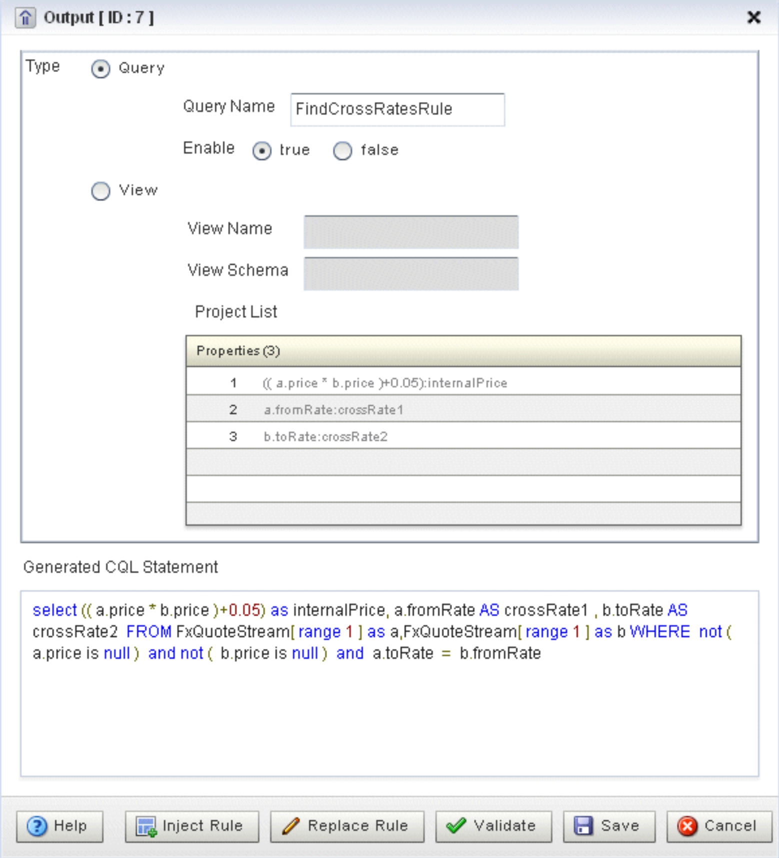
1. Double click the **Output** icon as Figure 6-73 shows.

   Figure 6-73 Output Icon

   The Output editor dialog appears as Figure 6-74 shows.

Figure 6-74 Query Wizard Output Editor Dialog



Type ● Query

Query Name FindCrossRatesRule

Enable ● true ○ false

○ View

View Name

View Schema

Project List

Properties (3)	
1	((a.price * b.price)+0.05):internalPrice
2	a.fromRate:crossRate1
3	b.toRate:crossRate2

Generated CQL Statement

```
select (( a.price * b.price )+0.05) as internalPrice, a.fromRate AS crossRate1 , b.toRate AS
crossRate2  FROM FxQuoteStream[ range 1 ] as a,FxQuoteStream[ range 1 ] as b WHERE  not (
a.price is null )  and not (  b.price is null )  and  a.toRate  =  b.fromRate
```

2. Edit the dialog as Table 6-22 describes.

Table 6-22 Query Wizard Output Editor Attributes

Attribute	Description
`Type`	The type of rule: • **Query**: An Oracle CQL query is an operation that you express in Oracle CQL syntax and execute on an Oracle CQL Processor to process data from one or more streams or views. • **View**: An Oracle CQL view represents an alternative selection on a stream or relation. In Oracle CQL, you use a view instead of a subquery.
`Query Name`	The name of the query.
`Enable`	Set this option to **true** to run the rule immediately after saving. Set this option to **false** to not run the rule after saving. A view cannot be enabled or disabled. If the query that uses a view is enabled, then the view is also enabled. If the query that uses a view is disabled, then the view is effectively disabled.
`View Name`	The name of the view.

Attribute	Description
`View Schema`	Editable list of the properties you specify in the `SELECT` clause. If you edit the generated CQL statement to add or remove properties from the `SELECT` clause, you must make the same change to the **View Schema**.
`Project List`	Read-only list of the properties you specify in the `SELECT` clause.
`Generated CQL Statement`	Editable Oracle CQL fragment associated with this CQL Construct.

3. To inject a new instance of the entire Oracle CQL rule into the Oracle CQL processor:

   a. If you are creating a new rule, Click **Inject Rule**.

      A new instance of the Oracle CQL rule is now visible on the Oracle Rules tab.

   b. If you are modifying an existing rule:

      - Click in the **Query Name** or **View Name** field and change the query or view name.

      - Click **Inject Rule**.

      A new instance of the Oracle CQL rule is now visible on the Oracle Rules tab with the new name.

4. To replace an existing instance of the entire Oracle CQL rule in the Oracle CQL processor, click **Replace Rule**.

   The existing instance of the Oracle CQL rule is updated on the Oracle Rules tab with the existing name.

5. Click the **Validate** button to ensure that your changes are syntactically correct.

   Correct any errors before proceeding.

6. Click **Save**.

   The Output icon displays a green check mark to indicate that it is configured.

6.1.9.17 Configure an Expression Using the Expression Builder

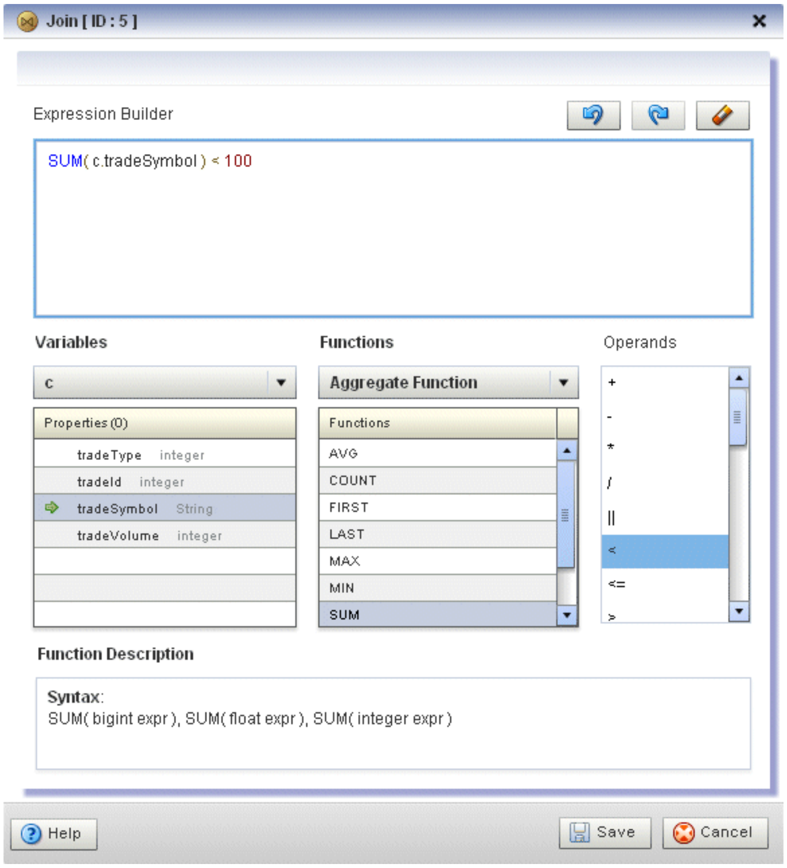
You can use the Query Wizard Expression Builder to construct an expression using various Oracle CQL operators, expressions, built-in single-row and aggregate functions, and user-defined functions.

Configure an Expression Using the Expression Builder

1. Click the **Expression Builder** button on the CQl construct editor dialog.

   The Expression Builder dialog appears. Figure 6-75 shows the Expression Builder after clicking the Expression Builder button in the Pattern construct editor.

Figure 6-75 Query Wizard Expression Builder



Join [ID : 5]

Expression Builder

```
SUM( c.tradeSymbol ) < 100
```

Variables: c

Functions: Aggregate Function

Properties (0)	
tradeType	integer
tradeId	integer
tradeSymbol	String
tradeVolume	integer

Function Description

Syntax:
SUM(bigint expr), SUM(float expr), SUM(integer expr)

2. Edit the dialog as Table 6-23 describes.

Table 6-23 Query Wizard Expression Builder Attributes

Attribute	Description
`Expression Builder`	Editable field that contains the expression.
`Variables`	Select a source from the **Variables** pull-down menu to list the properties it provides in the Properties list. Double-click a property to select it and add it to the Expression Builder at the current insertion point.
`Functions`	Select a function category from the **Functions** pull-down menu to list the functions it provides in the **Functions** list. Select a function to see its syntax in the Function Description field. Double-click a function to select it and add it to the Expression Builder at the current insertion point.
`Operands`	Double-click an operand to select it and add it to the Expression Builder at the current insertion point.
`Function Description`	Read-only field that shows the syntax for the currently selected function.

3. Select a source from the **Variables** pull-down menu.

4. Select a function category from the **Functions** pull-down menu.

5. Double-click a function to select it and add it to the Expression Builder.

6. Double-click a property to select it and make it the argument of the function.

7. Use your mouse to position the insertion bar after the closing bracket of the function.

8. Double click an operand to select it and add it to the Expression Builder.

9. Double-click another property or type in a literal value in the Expression Builder.

10. To undo an operation, click the **Undo** button.

11. To redo an operation, click the **Redo** button.

12. To clear the Expression Builder, click the **Erase** button.

13. Click **Save** to save the expression and return to the CQL construct editor dialog.

6.1.10 Create an Oracle CQL Template for the Query Wizard

Oracle Event Processing Visualizer provides a sophisticated Query Wizard to simplify Oracle CQL view and query construction. You can drag and drop a complete Oracle CQL view or query template and customize it to suit your needs. This procedure describes how to create your own templates and add them to the Query Wizard **Templates** palette.

An Oracle CQL template is based on the `wlevs_queryconstructor_config.xsd` from Oracle CQL Query Wizard Template Schema Reference. However, a user-defined template may contain only the following:

- `select-block`
- `from-block`
- `where-block`
- `Operator` element and attributes `ID` and `type`
- `Operator` element `inputs` child element

For more information on using templates to create an Oracle CQL rule, see Create a Rule with the Query Wizard.

Create an Oracle CQL Template for the Query Wizard:

1. Using your favorite editor, create an XML file and add the header.

```
<?xml version="1.0" encoding="UTF-8"?>
```

2. Create a `Rule` element.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule>
    <select-block>

    </select-block>

    <from-block>
    </from-block>

    <where-block>

    </where-block>


</Rule>
```

3. Add `Operator` elements to your template.

   Configure each `Operator` with a unique `ID` and specify the `type` attribute.

   Configure the `Rule` element `root` attribute with the `ID` of the operator that is the root (or ending point) of your rule. In the following example, the `Operator` of type `Output` with the `ID` 10 is the root of the rule.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule root="10">
    <select-block>

        <Operator ID="10" type="Output">
        </Operator>

        <Operator ID="9" type="IStream">
        </Operator>

        <Operator ID="7" type="Select">
        </Operator>

    </select-block>

    <from-block>

        <Operator ID="1" type="SSource">
        </Operator>

        <Operator ID="2" type="Window">
        </Operator>

        <Operator ID="3" type="SSource">
        </Operator>

        <Operator ID="4" type="Window">
        </Operator>

    </from-block>

    <where-block>

        <Operator ID="5" type="Join">
        </Operator>

    </where-block>
</Rule>
```

4. Connect your operators together by adding `inputs` elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<Rule root="9">
    <select-block>

        <Operator ID="10" type="Output">
            <inputs>
                <input>9</input>
            </inputs>
        </Operator>
        <Operator ID="9" type="IStream">
            <inputs>
                <input>8</input>
            </inputs>
        </Operator>

        <Operator ID="7" type="Select">
            <inputs>
                <input>5</input>
            </inputs>
        </Operator>

    </select-block>

    <from-block>

        <Operator ID="1" type="SSource">
        </Operator>

        <Operator ID="2" type="Window">
            <inputs>
                <input>1</input>
            </inputs>
        </Operator>

        <Operator ID="3" type="SSource">
        </Operator>

        <Operator ID="4" type="Window">
            <inputs>
                <input>3</input>
            </inputs>
        </Operator>

    </from-block>

    <where-block>

        <Operator ID="5" type="Join">
            <inputs>
                <input>2</input>
                <input>4</input>
            </inputs>
        </Operator>

    </where-block>

</Rule>
```

5. Save and close your template XML file.

Note:

If you enter double-byte characters in your template XML file, ensure that you save the file using UTF-8 encoding. For more information, see Configuration File Encoding: UTF-8.

6. Copy your template XML file to the `*ORACLE_CEP_HOME*/user_projects/domains/*DOMAIN_DIR*/*servername*/cqltemplate` directory of your Oracle Event Processing server, where *`ORACLE_CEP_HOME`* refers to the Oracle Event Processing installation directory such as `d:\oracle_cep`, *`DOMAIN_DIR`* refers to the name of your domain directory, and *`servername`* refers to the name of your server. For example: `d:\oracle_cep_home\user_projects\domains\mydomain\myserver1\cqltemplate`.

7. Add the name of your template XML file to the *ORACLE_CEP_HOME/* `user_projects/domains/`*DOMAIN_DIR*`/`*servername*`/cqltemplate/registry.xml` file.

   The following example shows how to register the template XML file `example.xml`.

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <templates>
       <template>
           <name>example</name>
           <filename>example.xml</filename>
           <description>This is an example user-defined cql statement</description>
       </template>
   </templates>
   ```

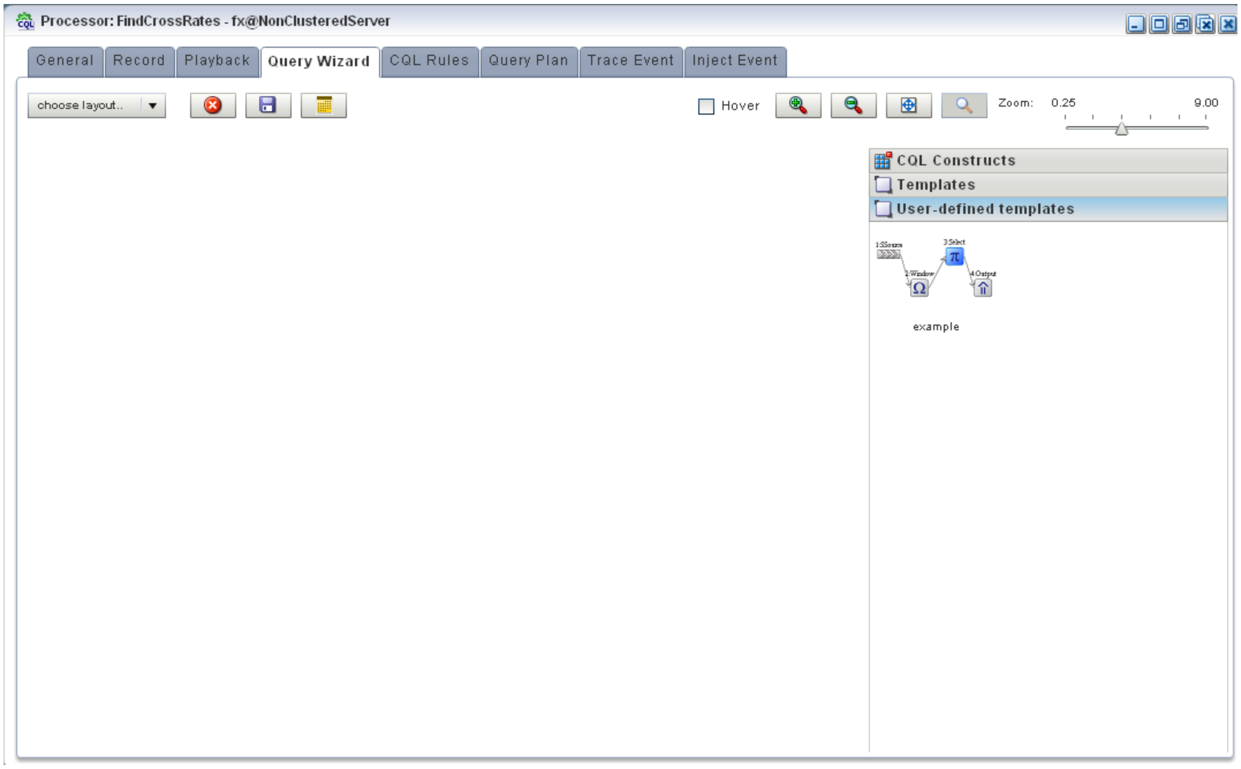
   The `template` element `name` child element determines the title that the Query Wizard displays in the **User-defined templates** tab for this template.

8. Exit out of the Oracle Event Processing Visualizer and log back in.

   For more information, see Start Visualizer.

9. Open the Oracle CQL Query Wizard and locate your template in the **User-defined templates** tab as Figure 6-7 shows.

Figure 6-76 Query Wizard User-defined templates Tab

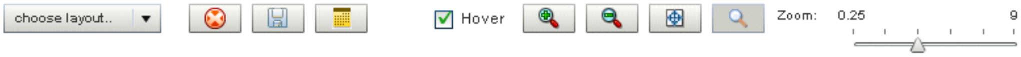


For more information, see Create a Rule with the Query Wizard.

6.1.11 Manage the Query Wizard Diagram

This section describes the various tools along the top of the Query Wizard canvas as Figure 6-77 shows.

Figure 6-77 Query Wizard Tools



You use these tools to manage the Query Wizard diagram of your Oracle CQL statement.

6.1.11.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing CQL constructs and showing their dependencies. You can choose any of:

- Left-Right
- Bottom-Top
- Right-Left
- Top-Bottom

6.1.11.2 Clear Canvas

Click the **Clear Canvas** button to erase the current Oracle CQL statement and its diagram from the Query Wizard canvas.

6.1.11.3 Save Query

Click the **Save Query** button at any time (even if you have not yet completed your Oracle CQL statement) to save your work to your local disk instead of the host.

6.1.11.4 Open Query

Click the **Open Query** button to reload a previously saved query.

6.1.11.5 Hover

Check the **Hover** option to display the Oracle CQL statement fragment associated with a given stage when you hover your mouse pointer over that stage.

Uncheck the **Hover** option to disable this feature.

6.1.11.6 Zoom In and Zoom Out

Click the **Zoom In** and **Zoom Out** buttons to change the zoom level. This is an alternative to using the Zoom slider (see Zoom).

6.1.11.7 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

6.1.11.8 Toggle Constructs

Click the **Toggle** Constructs button to alternately hide and show the CQL Constructs and Templates gallery.

6.1.11.9 Zoom

Use the Zoom slider to increase or decrease the zoom level. This is an alternative to using the Zoom In and Zoom Out buttons (see Zoom In and Zoom Out).

6.2 Use the Query Plan

Oracle Event Processing Visualizer provides a sophisticated Query Plan facility to simplify Oracle CQL query optimization.

Using the Query Plan facility, you can decompose a given Oracle CQL processor into its internal operators, states, and synopsis and you can gather various statistics on these operators such as incoming and outgoing number of messages. The Query Plan facility generates one query plan per Oracle CQL processor and applies to all the queries and views you define on that Oracle CQL processor.

The Oracle CQL Query Plan facility is designed to allow system administrators to efficiently diagnose and optimize Oracle CQL query performance.

This section describes the following topics:

- View a Query Plan for an Oracle CQL Processor

> **Note:**
>
> Before Oracle Event Processing Visualizer can generate a query plan, there must be at least one running Oracle CQL query (with the **Running** attribute set to `true`) on the Oracle CQL processor. For more information, see View a Rule.

6.2.1 View a Query Plan for an Oracle CQL Processor

You can view a query plan for an Oracle CQL processor using the Oracle Event Processing Visualizer.

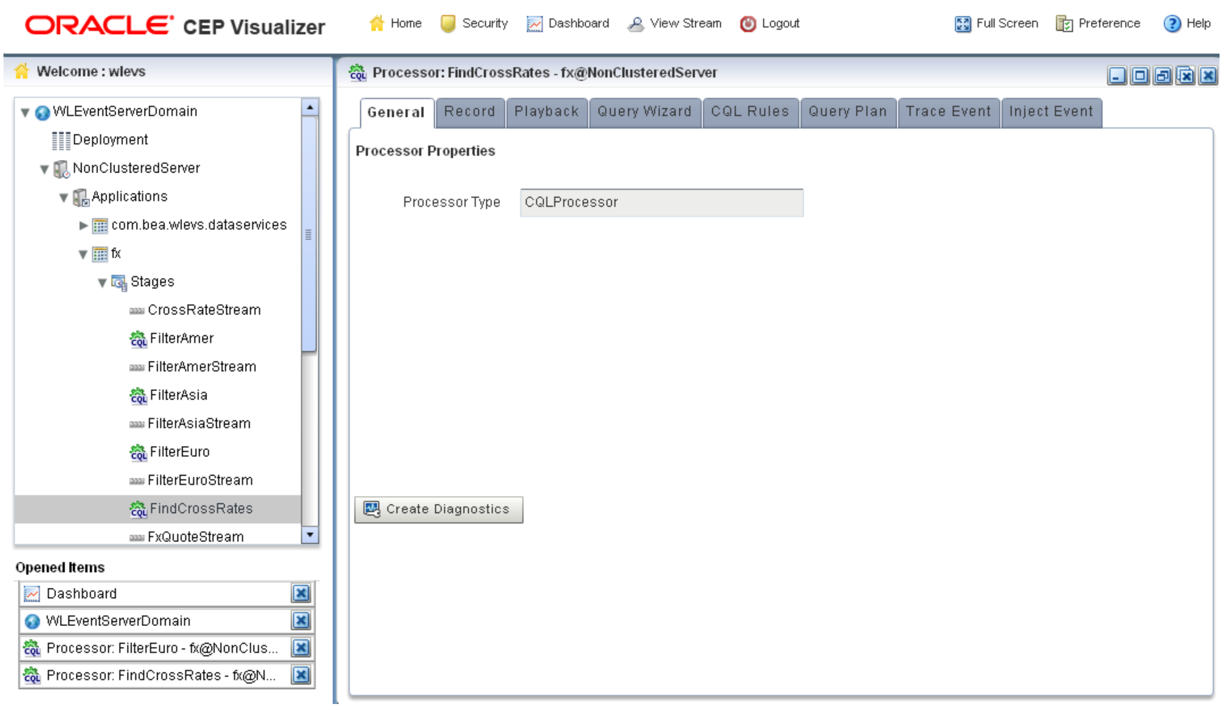
View a Query Plan for an Oracle CQL Processor

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application you want to use.
3. Select the Oracle CQL processor you wish to use:
   a. To use the EPN diagram:
      - Click the **Event Processing Network** tab.
      - Double-click the Oracle CQL processor you wish to use.

   b. To use the domain tree:
      - Expand the ***appname*** > **Stages** node, where ***appname*** is the name of the application you want to use.
      - Click the Oracle CQL processor you wish to use.
      - In the right pane, click the **General** tab

   The Processor panel is displayed as Figure 6-78 shows.

Figure 6-78 Oracle CQL Processor Panel



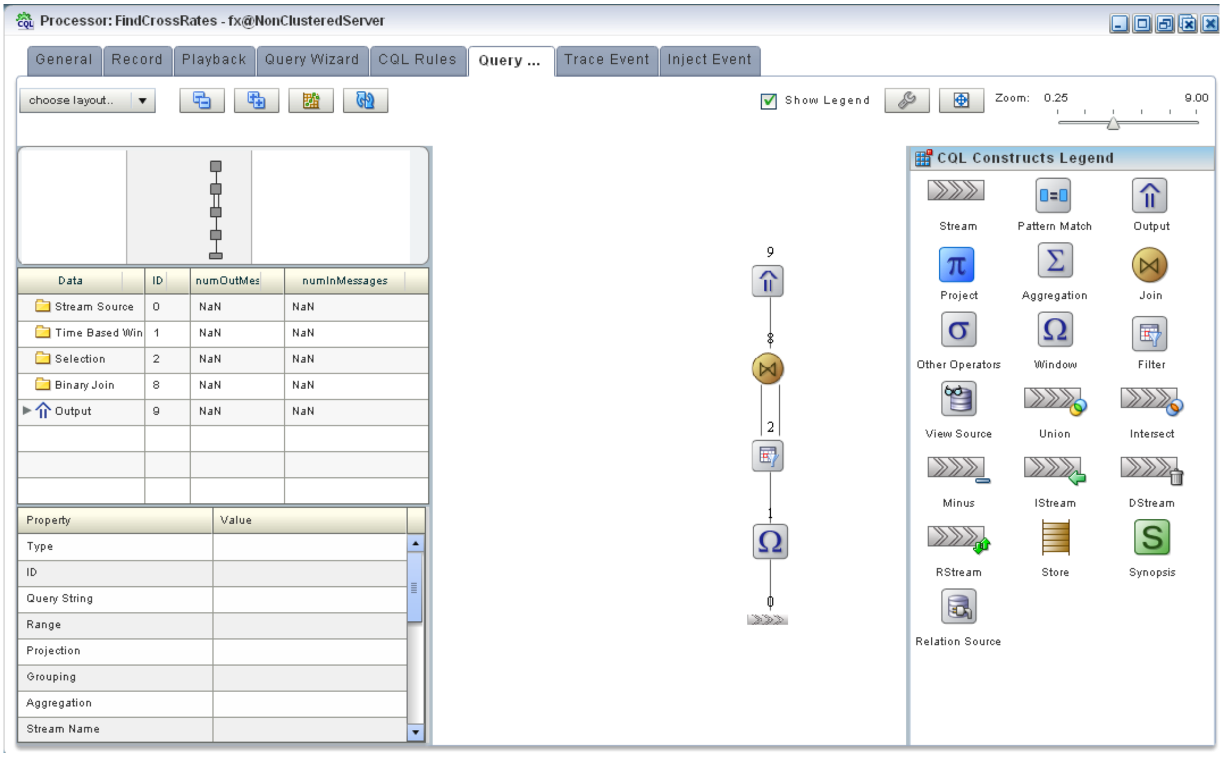
Processor Properties

Processor Type: CQLProcessor

4. Click the **Query Plan** tab.

   The Query Plan tab appears as Figure 6-79 shows.

Figure 6-79 Query Plan Tab



☑ Show Legend

Data	ID	numOutMes	numInMessages
Stream Source	0	NaN	NaN
Time Based Win	1	NaN	NaN
Selection	2	NaN	NaN
Binary Join	8	NaN	NaN
Output	9	NaN	NaN

Property	Value
Type	
ID	
Query String	
Range	
Projection	
Grouping	
Aggregation	
Stream Name	

For information on the various tools along the top of the Query Plan canvas, see Manage the Query Plan Diagram.

5. To view properties and values in the Property table for any stage, do either of the following:

   a. Click the stage icon in the query plan diagram on the right.

      The corresponding row in the Data table on the left is also selected.

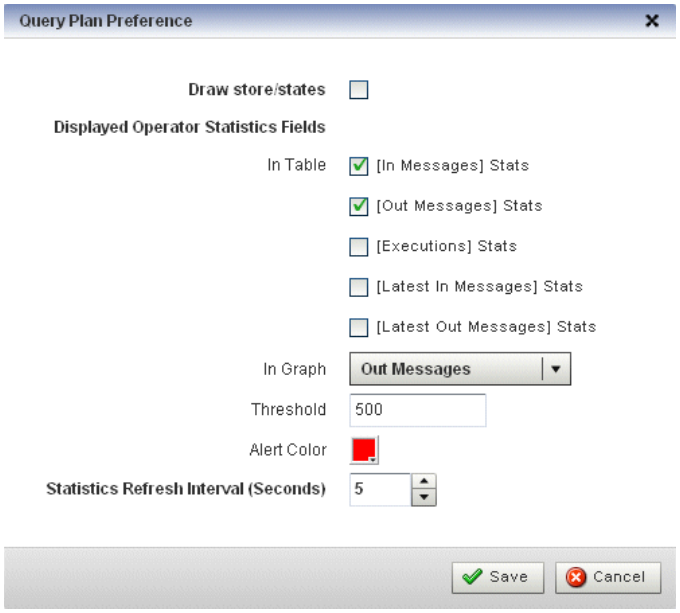
   b. Click the row in the Data table on the left that corresponds to the stage.

      The corresponding stage icon in the query plan diagram is also selected.

6. To configure the query plan, click the Query Plan Preference button.

   The Query Plan Preference dialog appears as Figure 6-80 shows.

Figure 6-80 Query Plan Preference Dialog



Draw store/states □

Displayed Operator Statistics Fields

In Table ☑ [In Messages] Stats

☑ [Out Messages] Stats

□ [Executions] Stats

□ [Latest In Messages] Stats

□ [Latest Out Messages] Stats

In Graph Out Messages ▾

Threshold 500

Alert Color 🟥

Statistics Refresh Interval (Seconds) 5

7. Edit the dialog as Table 6-24 describes.

Table 6-24 Query Plan Preference Attributes

Attribute	Description
`Draw store/ states`	Check this option to have Oracle Event Processing display store and states in the query plan diagram.

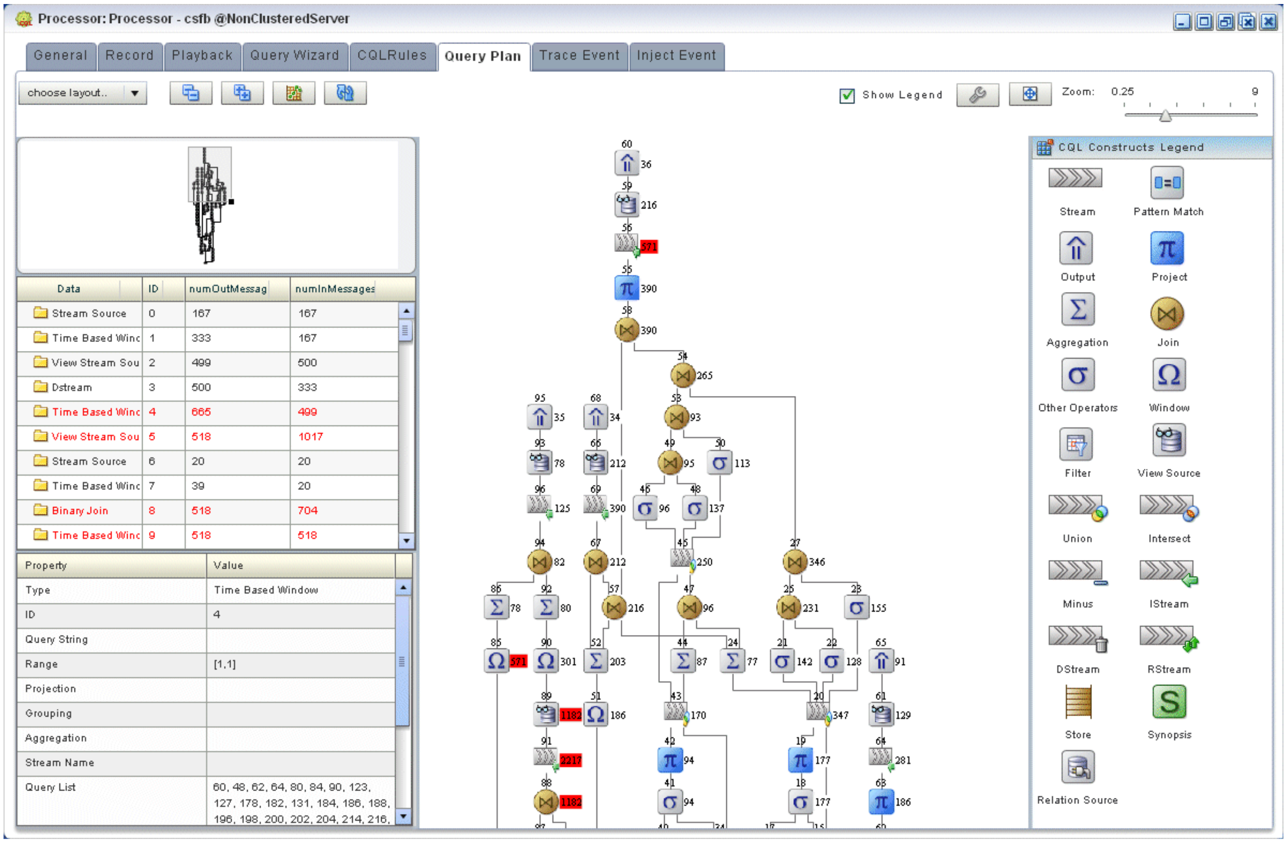
Attribute	Description
`Displayed Operator Statistics Fields`	Check the statistics you want to see in the Data table on the Query Plan tab. To configure the statistics displayed in the table, select: • **In Messages**: the total number of events received by this stage from its inbound stage. • **Out Messages**: the total number of events transmitted by this stage to its outbound stage. • **Executions**: the number of times an Oracle CQL query was executed on this stage. • **Latest In Messages**: the number of messages received from all input queues during the last invocation. • **Latest out Messages**: the number of tuples output by this operator during the last invocation. To configure the statistics displayed in the graph, select: • **NumInMessages**: the total number of events received by this stage from its inbound stage. • **NumOutMessages**: the total number of events transmitted by this stage to its outbound stage. • **NumExecutions**: the number of times an Oracle CQL query was executed on this stage. • **NumInMessagesLatest**: the number of messages received from all input queues during the last invocation. • **NumOutMessagesLatest**: the number of tuples output by this operator during the last invocation. **Note**: The **Threshold** value applies to the statistic you select here
`Threshold`	The maximum value for the Displayed Operator Stats Field in Graph statistic. If a stage exceeds this threshold, the value is colored on the query plan diagram using the color you specify for **Alert Color**. Default: 500.
`Alert Color`	Choose the color used to highlight stages that violate their threshold in the query plan display. If you do not want to show violations in color, select white as the alert color. Default: red.
`Statistics Refresh Interval` (Seconds).	Specify the frequency at which Oracle Event Processing collects statistics (in seconds). Default: 5.

8. To start gathering statistics, click the **Get Stats** button.

   The Get Stats button turns into a Stop Stats button.

   The Data table columns display the current statistics (according to your Displayed Operator Stats Fields in Table preference configuration) and threshold violations (according to your Displayed Operator Stats Field in Graph and Threshold preference configuration) are shown in the query plan diagram in red as Figure 6-81 shows.

Figure 6-81 Query Plan While Getting Statistics



Data	ID	numOutMessag	numInMessages
Stream Source	0	167	167
Time Based Winc	1	333	167
View Stream Sou	2	499	500
Dstream	3	500	333
Time Based Winc	4	665	499
View Stream Sou	5	518	1017
Stream Source	6	20	20
Time Based Winc	7	39	20
Binary Join	8	518	704
Time Based Winc	9	518	518

Property	Value
Type	Time Based Window
ID	4
Query String	
Range	[1,1]
Projection	
Grouping	
Aggregation	
Stream Name	
Query List	60, 48, 62, 64, 80, 84, 90, 123, 127, 178, 182, 131, 184, 186, 188, 196, 198, 200, 202, 204, 214, 216,

9. To stop getting statistics, click the **Stop Stats** button.

10. To drill-down into a stage to determine rate determining steps, do either of the following:

    a. Click the stage icon in the query plan diagram on the right that shows a threshold violation.

       The corresponding row in the Data table on the left is also selected and

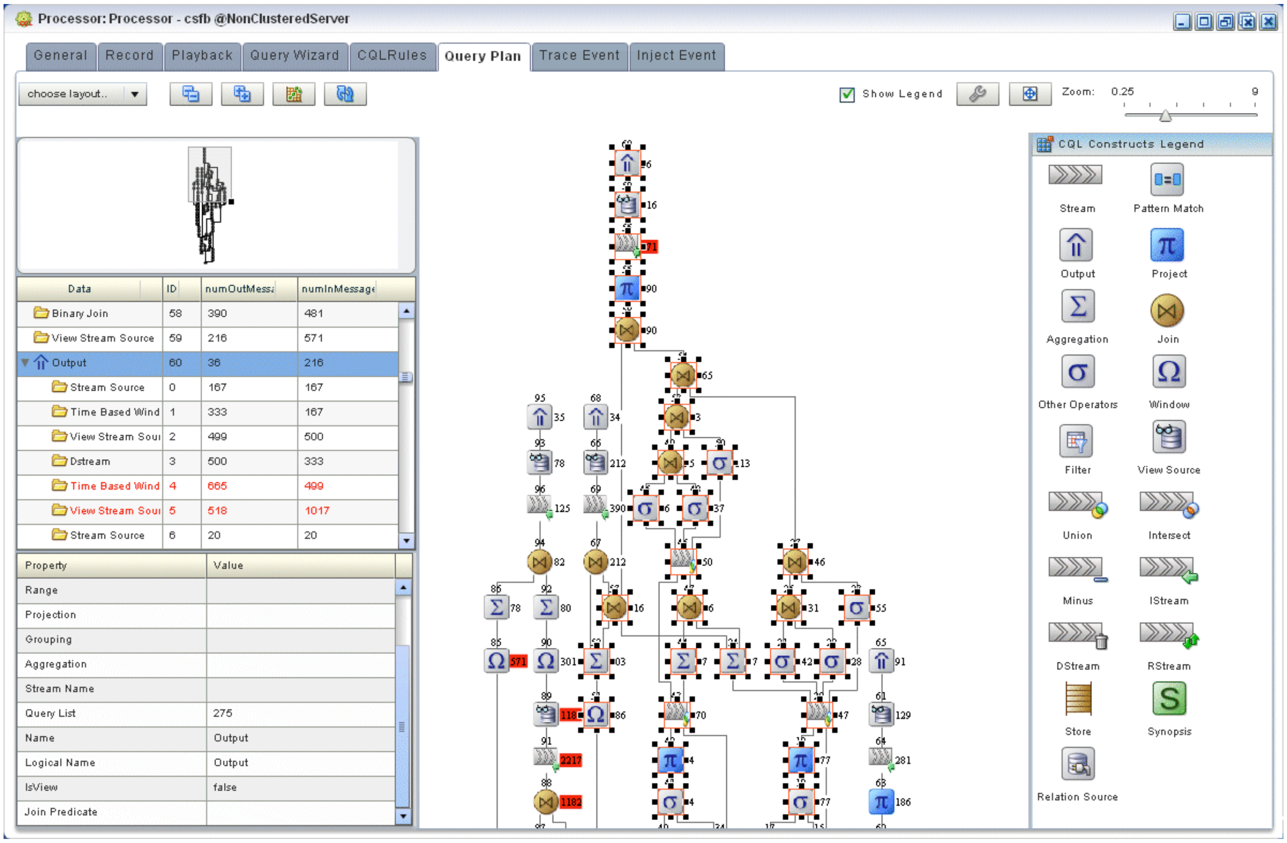
    b. Click the row in the Data table on the left that corresponds to the stage.

       The corresponding stage icon in the query plan diagram is also selected.

11. To view dependencies amongst stages, click on an Output row in the Data table.

    All the dependent stages are selected in the query plan diagram as Figure 6-82 shows.

Figure 6-82 Query Plan Showing Dependencies



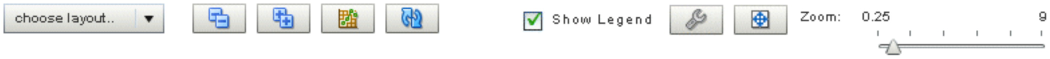
Data	ID	numOutMess:	numInMessage
Binary Join	58	390	481
View Stream Source	59	216	571
Output	60	36	216
Stream Source	0	167	167
Time Based Wind	1	333	167
View Stream Sour	2	499	500
Dstream	3	500	333
Time Based Wind	4	665	499
View Stream Sour	5	518	1017
Stream Source	6	20	20

Property	Value
Range	
Projection	
Grouping	
Aggregation	
Stream Name	
Query List	275
Name	Output
Logical Name	Output
IsView	false
Join Predicate	

6.2.2 Manage the Query Plan Diagram

This section describes the various tools along the top of the Query Wizard canvas as Figure 6-83 shows.

Figure 6-83 Query Plan Tools



You use these tools to manage the Query Wizard diagram of your Oracle CQL statement.

6.2.2.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing CQL constructs and showing their dependencies. You can choose any of:

- Left-Right
- Bottom-Top
- Right-Left
- Top-Bottom

6.2.2.2 Collapse All and Expand All

Use the **Collapse All** and **Expand All** buttons to collapse or expand the entries in the Data table.

6.2.2.3 Get Statistics and Stop Statistics

Use the **Get Statistics** button to instruct Oracle Event Processing to begin collecting statistics for the stages of the query plan.

When you click on the **Get Statistics** button, it turns into a **Stop Statistics** button. Click the **Stop Statistics** button to stop collecting statistics.

6.2.2.4 Refresh

Use the **Refresh** button to refresh the Query Plan diagram and Data table.

6.2.2.5 Show Legend

Use the **Show Legend** check box to show (checked) or hide (unchecked) the CQL Constructs Legend.

6.2.2.6 Preferences

Use the **Preferences** button to open the Query Plan pre-Renaissance dialog as Figure 6-80 shows.

6.2.2.7 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

6.2.2.8 Zoom

Use the Zoom slider to increase or decrease the zoom level.

7

Configuration History

This chapter describes how you can use Oracle Event Processing Visualizer to view, roll back, or delete configuration revision history, managing revisions by application or by resource.

This chapter includes the following sections:

- Manage Resource Configuration History
- Manage Application Configuration History

For more information, see Configuration History Management.

7.1 Manage Resource Configuration History

You can manage configuration history by resource as Resource Configuration History Management describes.

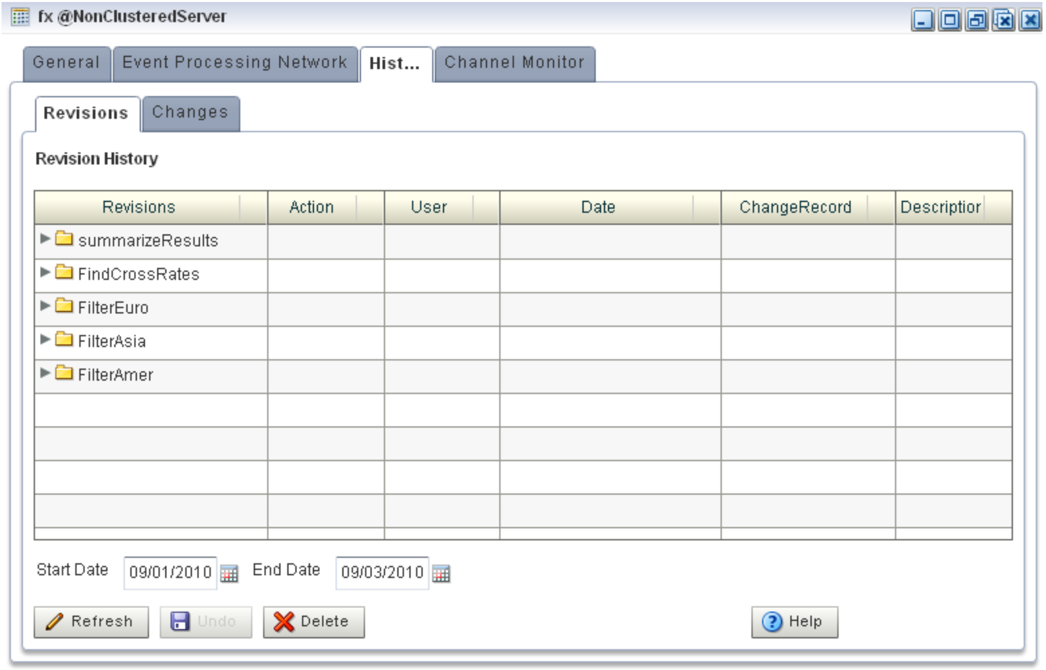
Alternatively, you can view configuration change history by application (see Manage Application Configuration History).

To manage resource configuration history:

1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.
2. Select ***appname***, where ***appname*** is the name of the application whose configuration history you want to manage.
3. In the right pane, click the **History** tab.

   The History tab is displayed as Figure 7-1 shows.

Figure 7-1 History Tab



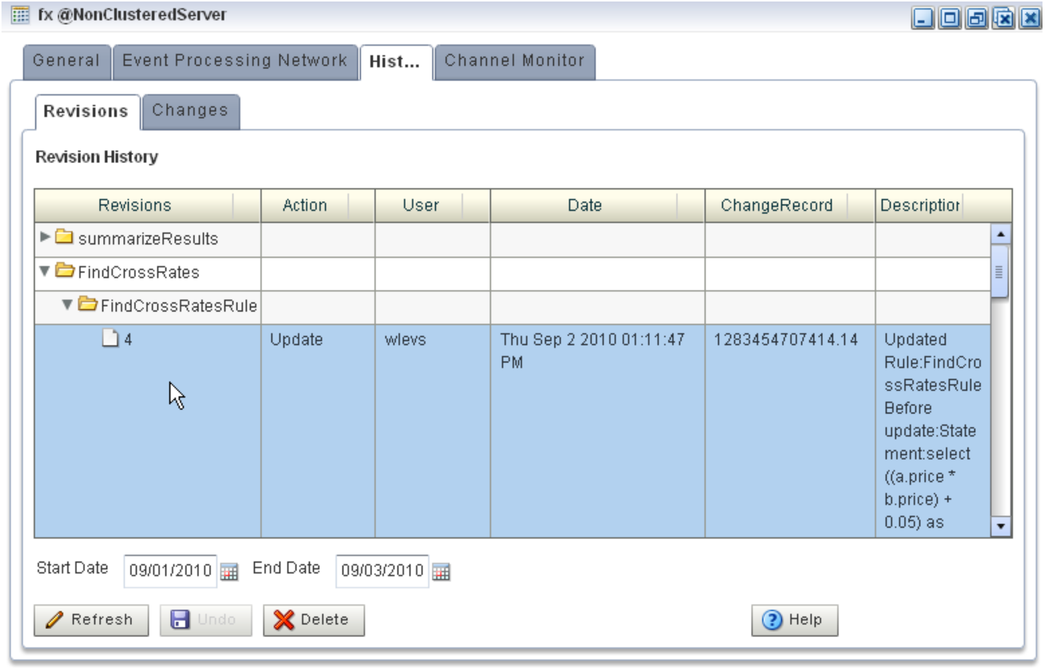
Revisions	Action	User	Date	ChangeRecord	Descriptior
▶ summarizeResults					
▶ FindCrossRates					
▶ FilterEuro					
▶ FilterAsia					
▶ FilterAmer					

Start Date 09/01/2010 End Date 09/03/2010

4. In the right pane, click the **Revisions** tab.

   The Revisions tab appears as Figure 7-2 shows.

Figure 7-2 Revisions Tab



Revisions	Action	User	Date	ChangeRecord	Descriptior
▶ summarizeResults					
▼ FindCrossRates					
▼ FindCrossRatesRule					
4	Update	wlevs	Thu Sep 2 2010 01:11:47 PM	1283454707414.14	Updated Rule:FindCrossRatesRule Before update:Statement:select ((a.price * b.price) + 0.05) as

Start Date 09/01/2010 End Date 09/03/2010

5. Filter the list of revisions by entering an appropriate **Start Time** and **End Time** and then click **Refresh**.

   The list refreshes to show only the revisions made in this time period.

6. Manage the resource change records:

   - To delete a revision, click **Delete**.

     All the revisions pertaining to the selected **Start Time** and **End Time** are deleted from the resource change history but resource configuration is unchanged.

   - To undo (roll back) the changes that the revision specifies, select each revision and click **Undo** until all of the revisions are rolled back.

     All the revisions pertaining to the selected **Start Time** and **End Time** are rolled back and the affected resources assume the configuration of the previous change record.

You can manage resource configuration history using the Oracle Event Processing Visualizer.

7.2 Manage Application Configuration History

You can manage configuration history by application as Application Configuration History Management describes. Alternately, you can manage configuration change history by resource (see Manage Resource Configuration History).

You can manage application configuration history using the Oracle Event Processing Visualizer.

To manage application configuration history:

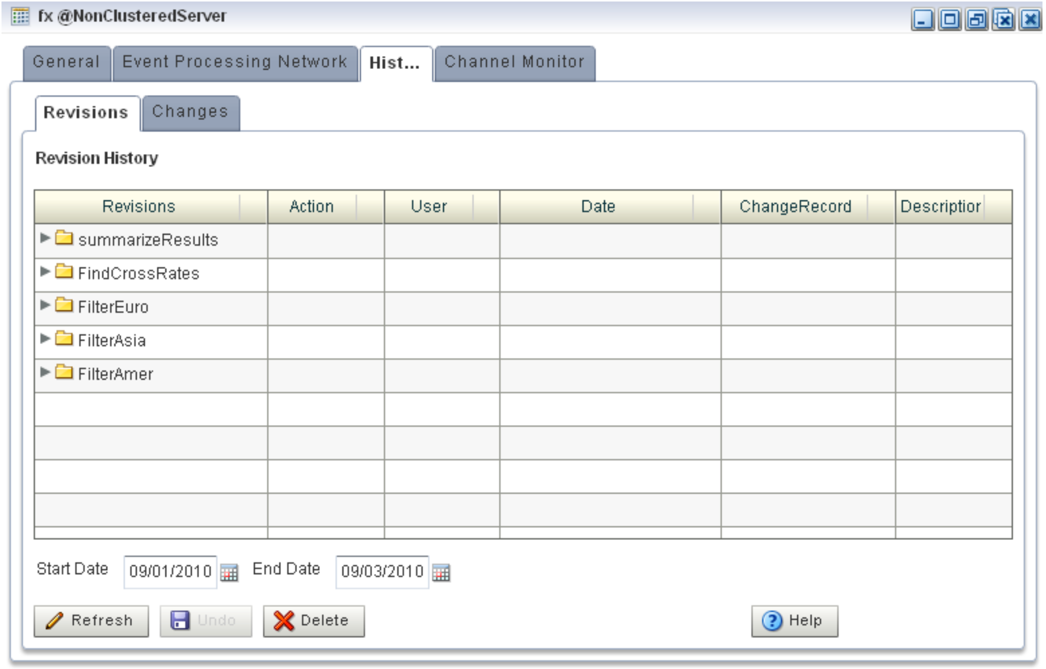
1. In the left pane, navigate to and expand the **Applications** node of the Oracle Event Processing instance to which the application is deployed.

2. Select ***appname***, where ***appname*** is the name of the application whose configuration history you want to manage.

3. In the right pane, click the **History** tab.

   The History tab is displayed as Figure 7-3 shows.

Figure 7-3 History Tab



Revision History

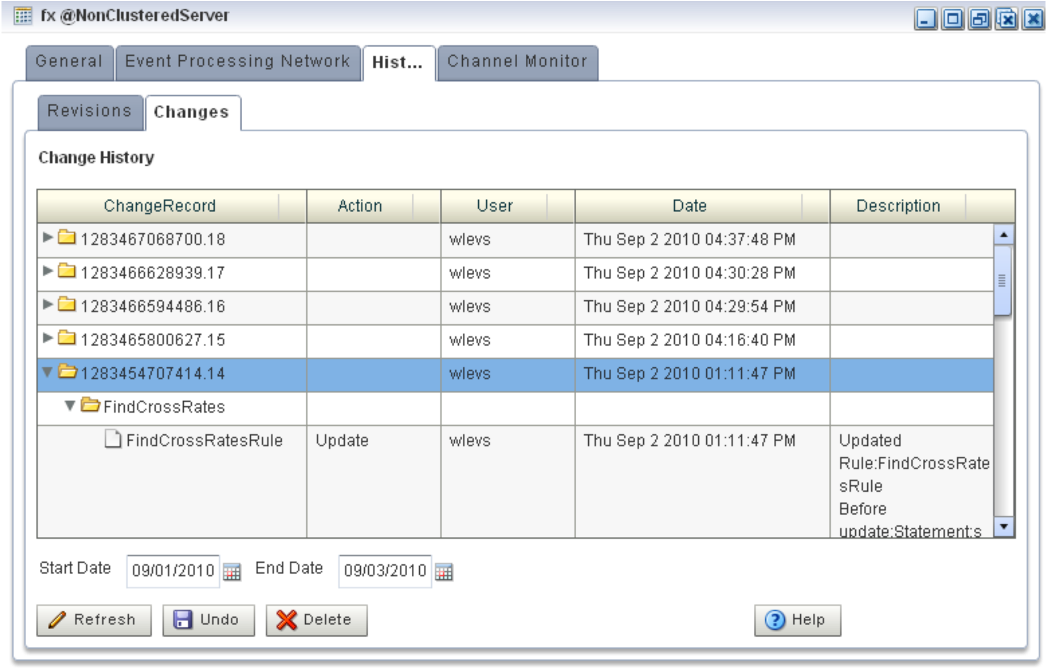
Revisions	Action	User	Date	ChangeRecord	Descriptior
▶ summarizeResults					
▶ FindCrossRates					
▶ FilterEuro					
▶ FilterAsia					
▶ FilterAmer					

Start Date 09/01/2010 End Date 09/03/2010

4. In the right pane, click the **Changes** tab.

   The Changes tab appears as Figure 7-4 shows.

Figure 7-4 Changes Tab



Change History

ChangeRecord	Action	User	Date	Description
▶ 1283467068700.18		wlevs	Thu Sep 2 2010 04:37:48 PM	
▶ 1283466628939.17		wlevs	Thu Sep 2 2010 04:30:28 PM	
▶ 1283466594486.16		wlevs	Thu Sep 2 2010 04:29:54 PM	
▶ 1283465800627.15		wlevs	Thu Sep 2 2010 04:16:40 PM	
▼ 1283454707414.14		wlevs	Thu Sep 2 2010 01:11:47 PM	
▼ FindCrossRates				
FindCrossRatesRule	Update	wlevs	Thu Sep 2 2010 01:11:47 PM	Updated Rule:FindCrossRatesRule Before update:Statement:s

Start Date 09/01/2010 End Date 09/03/2010

5. Filter the list of change records by entering an appropriate **Start Time** and **End Time** and then click **Refresh**.

   The list refreshes to show only the change records that were made in this time period.

6. Manage the application change records:

   - To delete change records, click **Delete**.

     All the change records pertaining to the selected **Start Time** and **End Time** are deleted from the application change history but the application configuration is unchanged.

   - To undo (roll back) the changes that a change record specifies, select each change record and click **Undo** until all of the change records are rolled back

     All the change records pertaining to the selected **Start Time** and **End Time** are rolled back and the affected application assumes the configuration of the previous change record.

Part III

Server and Domain Tasks

Server and Domain Tasks contains the following chapters:

- Server and Domain Tasks
- Application Deployments
- Jetty Configuration
- JMX Configuration
- Data Source Configuration
- HTTP Publish-Subscribe Servers
- Event Inspector Service
- Event Type Repository
- Persistent Event Store
- Multiserver Domains
- Server Logs

8

Server and Domain Tasks

This chapter introduces typical server and domain management tasks you can perform with Oracle Event Processing Visualizer, including managing deployment, multiserver domains, logs, caches, the event type repository, and more.

This chapter includes the following sections:

- Deployment Management
- Jetty Management
- JMX Management
- Data Source Management
- HTTP Publish-Subscribe Server Management
- Event Type Repository Management
- Persistent Event Store Management
- Multiserver Domain Management
- Event Inspector Service Management
- Cache Management
- Log Management.

8.1 Deployment Management

Using the Oracle Event Processing Visualizer, you can view the applications and application libraries deployed to a selected Oracle Event Processing server.

For more information, see:

- Application Deployments
- Application Lifecycle
- *Developing Applications for Oracle Event Processing*.

8.2 Jetty Management

Oracle Event Processing supports Jetty as the Java Web server to deploy HTTP servlets and static resources.

Oracle Event Processing support for Jetty is based on Version 1.2 the OSGi HTTP Service. This API provides the ability to dynamically register and unregister `javax.servlet.Servlet` objects with the run time and static resources.

Oracle Event Processing allows you to configure how your application prioritizes the execution of its work. Based on rules you define and by monitoring actual run time performance, you can optimize the performance of your application and maintain service level agreements. You define the rules and constraints for your application by defining a work manager.

For more information, see:

- Jetty Configuration
- *Administering Oracle Event Processing*
- *Developing Applications for Oracle Event Processing*.

8.3 JMX Management

Oracle Event Processing provides standards-based interfaces that are fully compliant with the Java Management Extensions (JMX) specification. Software vendors can use these interfaces to monitor Oracle Event Processing MBeans, to change the configuration of an Oracle Event Processing domain, and to monitor the distribution (activation) of those changes to all server instances in the domain.

Oracle Event Processing Visualizer and the `wlevs.Admin` command line tool both use JMX to connect to a server. However, to use these tools, and the JMX interfaces in general, you must configure Oracle Event Processing with the JMX configuration information in the `config.xml` file.

For more information, see:

- JMX Configuration
- *Administering Oracle Event Processing*.

8.4 Data Source Management

Oracle Event Processing supports Java Database Connectivity (JDBC) 3.0 for relational database access.

The JDBC API provides a standard, vendor-neutral mechanism for connecting to and interacting with database servers and other types of tabular resources that support the API. The JDBC `javax.sql.DataSource` interface specifies a database connection factory that is implemented by a driver. Instances of `DataSource` objects are used by applications to obtain database connections (instances of `java.sql.Connection`). After obtaining a connection, an application interacts with the resource by sending SQL commands and receiving results.

For more information, see:

- Data Source Configuration
- *Administering Oracle Event Processing*.

8.5 HTTP Publish-Subscribe Server Management

Oracle Event Processing includes an HTTP publish-subscribe server to which applications can publish messages. Applications publish messages to a particular channel; other applications can then subscribe to this channel to receive these published messages.

Oracle Event Processing HTTP pub-sub server channels may be:

- **Static:** those channels configured in the Oracle Event Processing server `config.xml` and used by Oracle Event Processing Visualizer itself.

  Oracle Event Processing Visualizer includes the following static channels (preconfigured in `config.xml`) that are used by Oracle Event Processing Visualizer itself; they cannot be deleted or modified:

  – `/evsmonitor`

  – `/evsalert`

  – `/evsdomainchange`

- **Dynamic:** those channels that are not configured in the Oracle Event Processing server `config.xml` but created dynamically when the Oracle Event Processing sever is running. You can view statistics for dynamic channels, including the number of messages sent to this channel and the number of subscribers. Dynamic channel configuration is read-only.

  Oracle Event Processing Visualizer supports only the Event Inspector service dynamic channel.

- **User-defined:** those static channels that your Oracle Event Processing application creates.

> **Note:**
>
> When one application (application-1) advertises a channel that a second application (application-2) subscribes to and uses as a foreign stage, you cannot view the details of the advertised channel from application-2. To view the details of this channel, you must do so from application-1.

You can use Oracle Event Processing Visualizer to modify existing channels, as well as create new ones and delete existing ones. When you configure channels with Oracle Event Processing Visualizer, they are permanent and survive server restart. Channels configured using APIs from a custom HTTP pub-sub adapter are dynamic and do not survive server restart.

For more information, see:

- HTTP Publish-Subscribe Servers
- HTTP Publish-Subscribe Server Security
- Trace and Inject Events in the EPN
- *Administering Oracle Event Processing*.

8.6 Event Type Repository Management

Event types define the properties of the events that are handled by Oracle Event Processing applications. All the event types used by the applications of a server make up the event type repository.

For more information, see:

- Event Type Repository
- *Developing Applications for Oracle Event Processing*.

8.7 Persistent Event Store Management

The Oracle Event Processing event repository feature allows you to persist the events that flow out of a component of the event processing network (EPN) to a store, such as a database table, and then play them back at a later stage or explicitly query the events from a component such as an event bean. By default, Oracle Event Processing stores recorded events in a database, which means that before you can start using the record and playback feature in your own application, you must specify where the database server is located along with the name of the database server that will contain the recorded events.

For more information, see:

- Persistent Event Store
- *Developing Applications for Oracle Event Processing*.

8.8 Multiserver Domain Management

An Oracle Event Processing multiserver domain (or cluster) is a set of two or more servers logically connected for the purposes of management, and physically connected using a shared User Datagram Protocol (UDP) multicast address and port. All servers in an Oracle Event Processing multiserver domain are aware of all other servers in the domain and any one server can be used as an access point for making changes to the deployments in the domain.

Oracle Event Processing supports the following clustering systems:

- evs4j: Oracle Event Processing native clustering implementation.
- Oracle Coherence: a JCache-compliant in-memory distributed data grid solution for clustered applications and application servers. It coordinates updates to the data using cluster-wide concurrency control, replicates data modifications across the cluster using the highest performing clustered protocol available, and delivers notifications of data modifications to any servers that request them. You take advantage of Oracle Coherence features using the standard Java collections API to access and modify data, and use the standard JavaBean event model to receive data change notifications.

Note:

Before you can use Oracle Event Processing with Oracle Coherence, you must obtain a valid Oracle Coherence license such as a license for Coherence Enterprise Edition, Coherence Grid Edition, or Oracle WebLogic Application Grid. For more information on Oracle Coherence, see `http://www.oracle.com/technetwork/middleware/coherence/overview/index.html`.

Note:

Any changes to rules and Oracle high availability adapters are propagated to the other servers in the same group. That is, all rule and Oracle high availability adapter configurations is automatically synchronized. Other configuration changes are not synchronized. For example, if you change record/playback or JMS adapter configuration on one server in a multiserver domain, then these changes are not synchronized with the other servers in the same group. For more information, see:

- Oracle CQL Rules
- View and Edit the Configuration of a Stage.

For more information, see:

- Multiserver Domains
- Multiserver Domains and Configuration History Management
- *Administering Oracle Event Processing*.

8.9 Event Inspector Service Management

Using the Event Inspector service, you can:

- View the events flowing out of any stage in the EPN
- Inject events into any stage in the EPN

You can use the Event Inspector service to test and debug Oracle CQL queries during development.

For more information, see:

- Event Inspector Service
- Trace and Inject Events in the EPN
- *Developing Applications for Oracle Event Processing*.

8.10 Cache Management

A *cache* is a temporary storage area for events, created exclusively to improve the overall performance of your Oracle Event Processing application; it is not necessary for the application to function correctly. Oracle Event Processing applications can optionally publish or consume events to and from a cache to increase the availability of the events and increase the performance of their applications.

A *caching system* refers to a configured instance of a caching implementation. A caching system defines a named set of configured caches as well as the configuration for remote communication if any of the caches are distributed across multiple machines.

Oracle Event Processing supports the following caching systems:

- Oracle Event Processing local cache: a local, in-memory single-JVM cache.

- Oracle Coherence: a JCache-compliant in-memory distributed data grid solution for clustered applications and application servers. It coordinates updates to the data using cluster-wide concurrency control, replicates data modifications across the cluster using the highest performing clustered protocol available, and delivers notifications of data modifications to any servers that request them. You take advantage of Oracle Coherence features using the standard Java collections API to access and modify data, and use the standard JavaBean event model to receive data change notifications.

  Note:

  Before you can use Oracle Event Processing with Oracle Coherence, you must obtain a valid Oracle Coherence license such as a license for Coherence Enterprise Edition, Coherence Grid Edition, or Oracle WebLogic Application Grid. For more information on Oracle Coherence, see `http://www.oracle.com/technetwork/middleware/coherence/overview/index.html`.

- Third-party caches: you can create a plug-in to allow Oracle Event Processing to work with other, third-party cache implementations.

For more information, see:

- Multiserver Domains
- *Developing Applications for Oracle Event Processing*.

8.11 Log Management

Oracle Event Processing Visualizer allows you to change the server-wide logging configuration, such as the severity of log messages you want the server to print out, the name of the log file, and so on.

You can also view and modify the logging level of individual components and configure logging for new components.

For more information, see:

- Server Logs
- *Administering Oracle Event Processing*.

9 Application Deployments

This chapter describes how you can use Oracle Event Processing Visualizer to view, add, or delete Oracle Event Processing application deployments on Oracle Event Processing servers.

This chapter includes the following sections:

- View the Applications Deployed to a Server
- View the Application Libraries Deployed to a Server
- Add or Delete an Application
- Add or Delete an Application Library

For more information, see:

- Deployment Management
- Application Lifecycle.

9.1 View the Applications Deployed to a Server

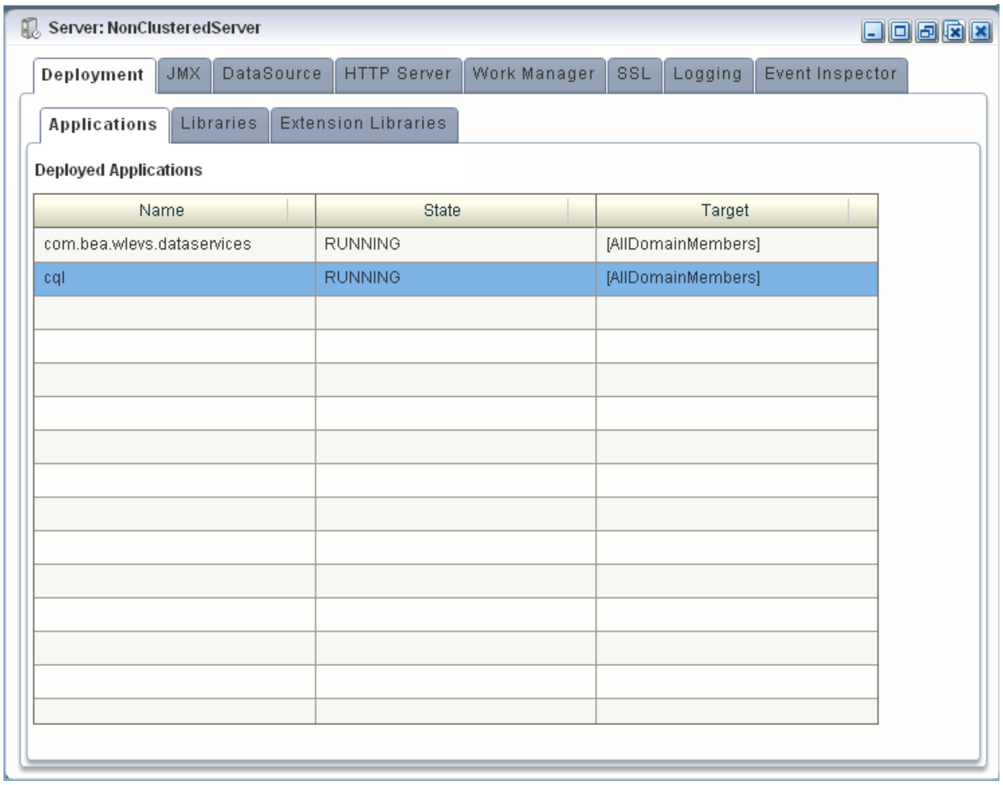
Using the Oracle Event Processing Visualizer, you can view the applications deployed to a selected Oracle Event Processing server.

View the Applications Deployed to a Server

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Applications** tab.

   The Applications tab is displayed as Figure 9-1 shows.

Figure 9-1 Applications Tab



Deployed Applications

Name	State	Target
com.bea.wlevs.dataservices	RUNNING	[AllDomainMembers]
cql	RUNNING	[AllDomainMembers]

3. View the list of deployed applications as Table 9-1 describes.

Table 9-1 Deployed Application Attributes

Attribute	Description
Name	The name of the application.
State	The operational state of the application.
Target	The group to which the application is deployed.

9.2 View the Application Libraries Deployed to a Server

Using the Oracle Event Processing Visualizer, you can view the application libraries deployed to a selected Oracle Event Processing server.

You can view:

- **Extension Libraries**: these libraries are deployed first along with the Oracle Event Processing server extension modules.

  See *Developing Applications for Oracle Event Processing*.

- **Libraries**: these libraries are deployed after the components in the library extensions directory but before any Oracle Event Processing applications

  See *Developing Applications for Oracle Event Processing*.

View the Application Libraries Deployed to a Server

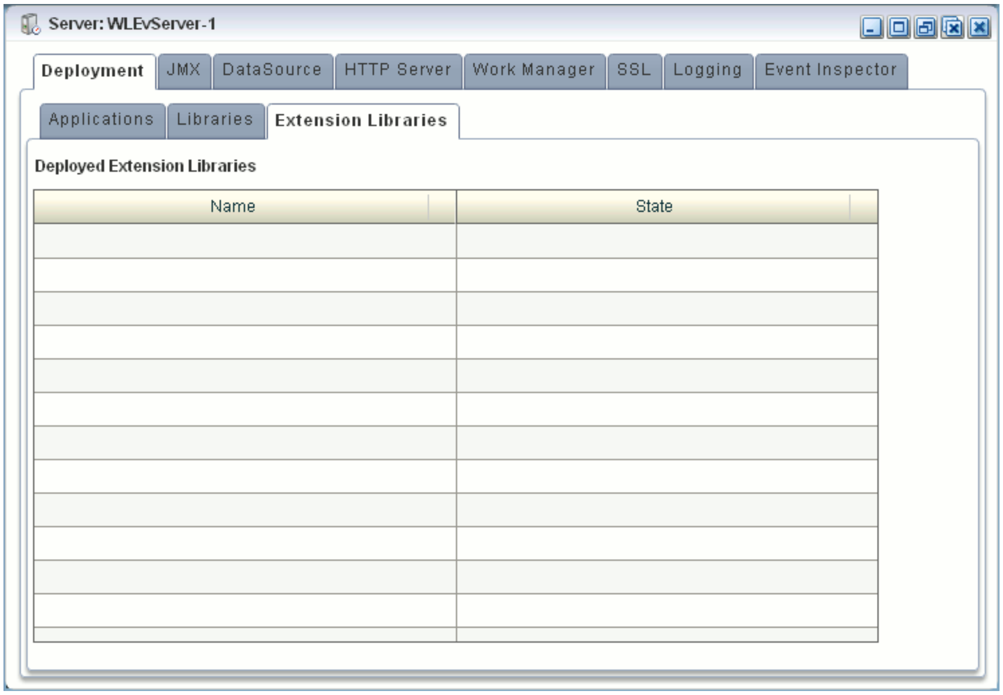
1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. View the libraries associated with applications deployed to this server:

   a. To view extension libraries, in the right pane, click the **Extension Libraries** tab.

      The Extension Libraries tab is displayed as Figure 9-2 shows.

      Figure 9-2 Extension Libraries Tab



      View the list of deployed application libraries as Table 9-2 describes.

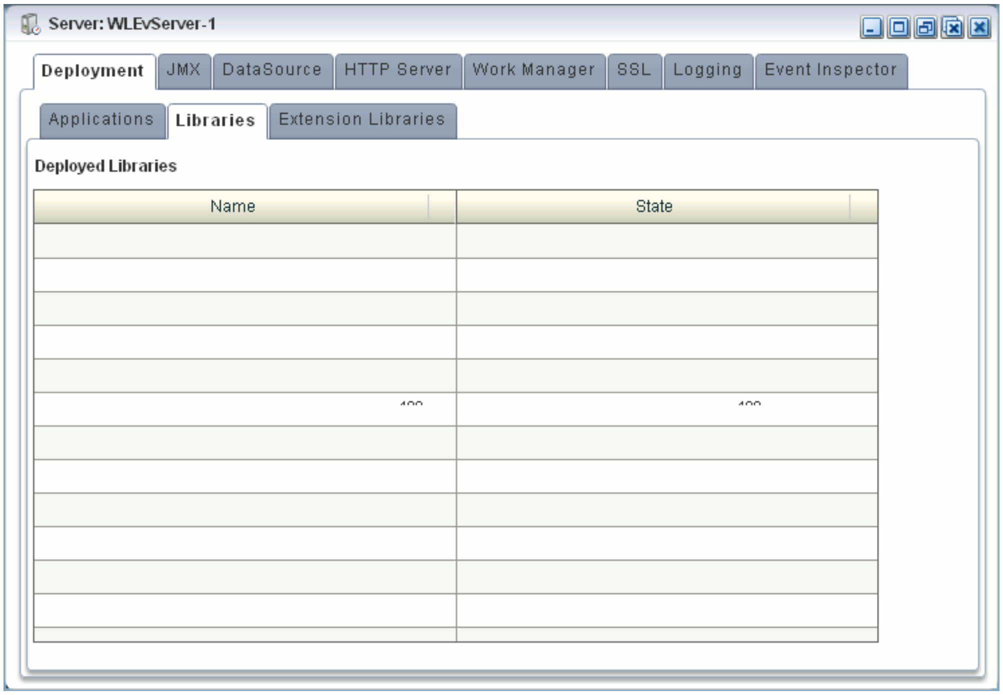
      Table 9-2 Application Extension Library Attributes

Attribute	Description
Name	The name of the application library.
State	The operational state of the application library.

   b. To view libraries, in the right pane, click the **Libraries** tab.

      The Libraries tab is displayed as Figure 9-3 shows.

Figure 9-3 Libraries Tab



View the list of deployed application libraries as Table 9-3 describes.

Table 9-3 Application Library Attributes

Attribute	Description
Name	The name of the application library.
State	The operational state of the application library.

9.3 Add or Delete an Application

For more information, see:

- Deploy an Application in a Standalone-Server Domain
- Deploy an Application in a Multiserver Domain
- Uninstall an Application.

9.4 Add or Delete an Application Library

To add or remove application libraries, you must manually add or remove files from the appropriate Oracle Event Processing server directory.

For more information, see *Developing Applications for Oracle Event Processing*.

10

Jetty Configuration

This chapter describes how you can use Oracle Event Processing Visualizer to view and edit Jetty web server configuration and how to view Jetty work manager configuration for an Oracle Event Processing server.

This chapter includes the following sections:

For more information, see Jetty Management.

10.1 View Jetty Web Server Configuration

Using the Oracle Event Processing Visualizer, you can view the Jetty Web server configuration for a selected Oracle Event Processing server.

View Jetty Web Server Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **HTTP Server** tab.
3. In the right pane, the properties of the Jetty server are displayed:
   - The name of the Jetty server.
   - The network I/O object to which the Jetty server is bound.
   - The work manager name to which the Jetty server is bound.
   - The list of Web applications that are deployed to this Jetty server, along with the path of the application and its context path.

10.2 Edit Jetty Web Server Configuration

You can only view the configuration of the Jetty servers configured in your Oracle Event Processing server using Oracle Event Processing Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

See *Administering Oracle Event Processing*.

10.3 View and Edit Work Manager Configuration

Using the Oracle Event Processing Visualizer, you can view the work manager configuration for a selected Oracle Event Processing server.

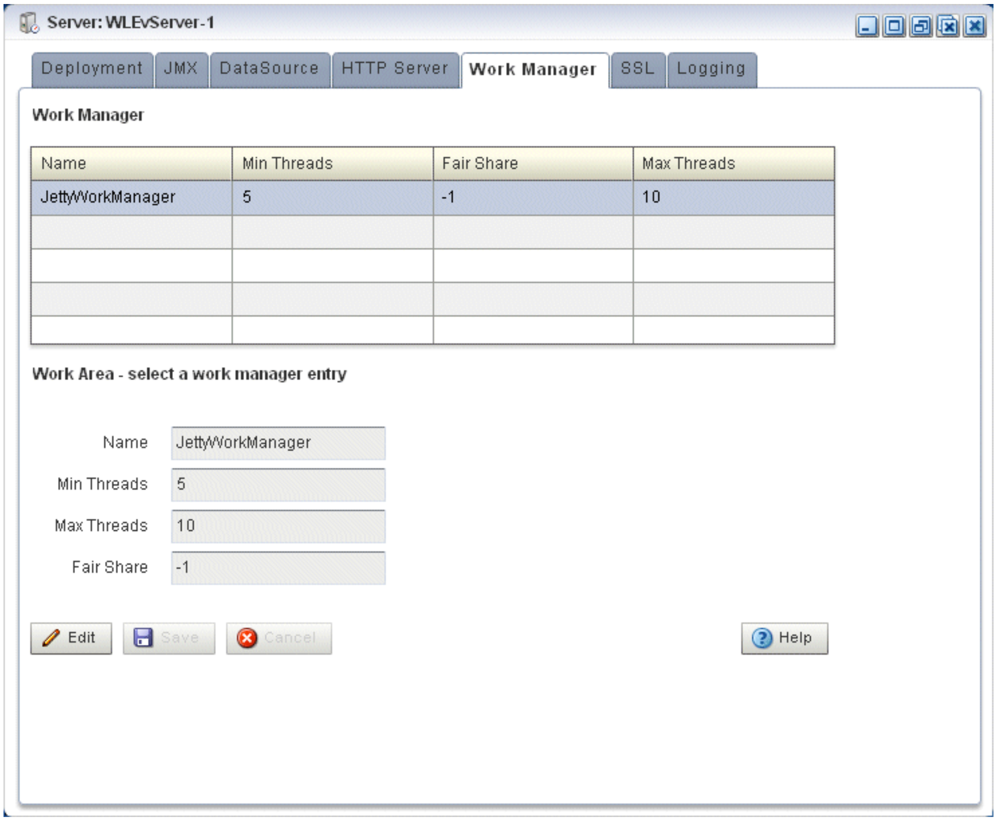
View the Work Manager Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. In the right pane, click the **Work Manager** tab.

   The Work Manager tab appears as shown in Figure 10-1.

Figure 10-1 Work Manager Tab



Work Manager

Name	Min Threads	Fair Share	Max Threads
JettyWorkManager	5	-1	10

Work Area - select a work manager entry

Name	JettyWorkManager
Min Threads	5
Max Threads	10
Fair Share	-1

3. In the Work Manager table, select the work manager you want to configure by clicking on its name.

4. Click the **Edit** button at the bottom of the page.

5. Update the work manager configuration properties:

   - `Min threads`—The minimum number of threads that the server allocates so as to reduce the initial time it takes to perform a task.

   - `Max threads`—The maximum number of concurrent threads that execute requests from the constrained work set.

- `Fair Share`—The average thread-use time required to process requests.

6. Click **Save** to save your changes or Cancel to cancel the update.

10.4 Create a Work Manager

You cannot delete existing work managers or create new ones using Oracle Event Processing Visualizer; rather, you must manually update the server's `config.xml` file. See *Administering Oracle Event Processing*.

10.5 Delete a Work Manager

You cannot delete existing work managers or create new ones using Oracle Event Processing Visualizer; rather, you must manually update the server's `config.xml` file. See *Administering Oracle Event Processing*.

11

JMX Configuration

This chapter describes how you can use Oracle Event Processing Visualizer to view Java Management Extensions (JMX) configuration for use with Oracle Event Processing.

This chapter includes the following sections:

- View JMX Configuration
- Editing JMX Configuration

For more information, see JMX Management.

11.1 View JMX Configuration

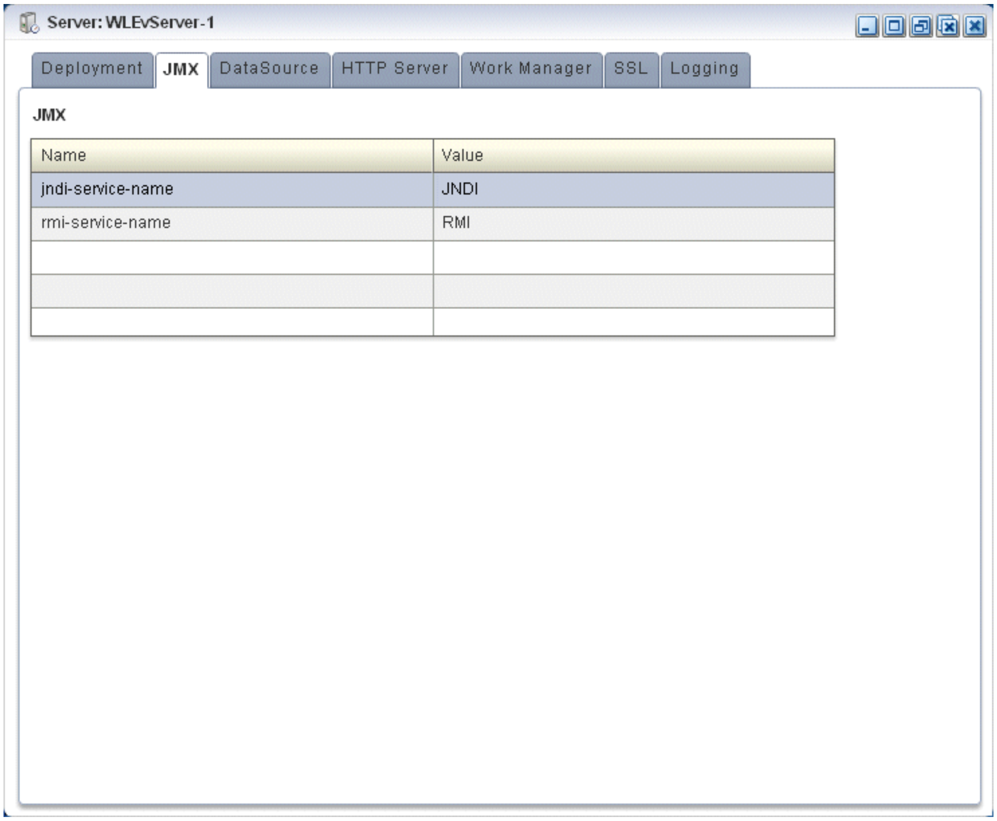
Using the Oracle Event Processing Visualizer, you can view the JMX configuration for a selected Oracle Event Processing server.

View JMX Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **JMX** tab.

   The JMX tab appears as Figure 11-1 shows.

Figure 11-1 JMX Tab



Name	Value
jndi-service-name	JNDI
rmi-service-name	RMI

3. The JMX properties are displayed in the table:

   - `jndi-service-name`—The name of the JNDI service to which the JMX server will bind its object.
   - `rmi-service-name`—The name of the RMI service with which the JMX server will register to receive calls.
   - `rmi-jrmp-port`—The port on which to listen for RMI JRMP JMX requests.
   - `rmi-registry-port`—The port on which to start the RMIRegistry.

Oracle Event Processing does not support the JRMP protocol. An Oracle Event Processing JMX client does not use the `rmi-jrmp-port` or `rmi-registry-port` properties. Instead, JMX clients must use the more secure MSA protocol for both local and remote access to the Oracle Event Processing JMX server.

When you connect to the Oracle Event Processing JMX server that is running on `localhost` or on a remote host, you must always use the JMX URL `service:jmx:msarmi://`*`HOST-NAME`*`:`*`PORT`*`/jndi/jmxconnector` so that you are always using the MSA connector (where *`HOST-NAME`* is either `localhost` or the name of the remote host and *`PORT`* is the Oracle Event Processing server JNDI port).

11.2 Editing JMX Configuration

You can only view the JMX configuration of your Oracle Event Processing server using Oracle Event Processing Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

See *Administering Oracle Event Processing*.

12

Data Source Configuration

This chapter describes how you can use Oracle Event Processing Visualizer to view, edit, and add data source configuration for use with Oracle Event Processing.

This chapter includes the following sections:

- View a Data Source Configuration
- Edit Data Source Configuration
- Add a Data Source Configuration

For more information, see Data Source Management.

12.1 View a Data Source Configuration

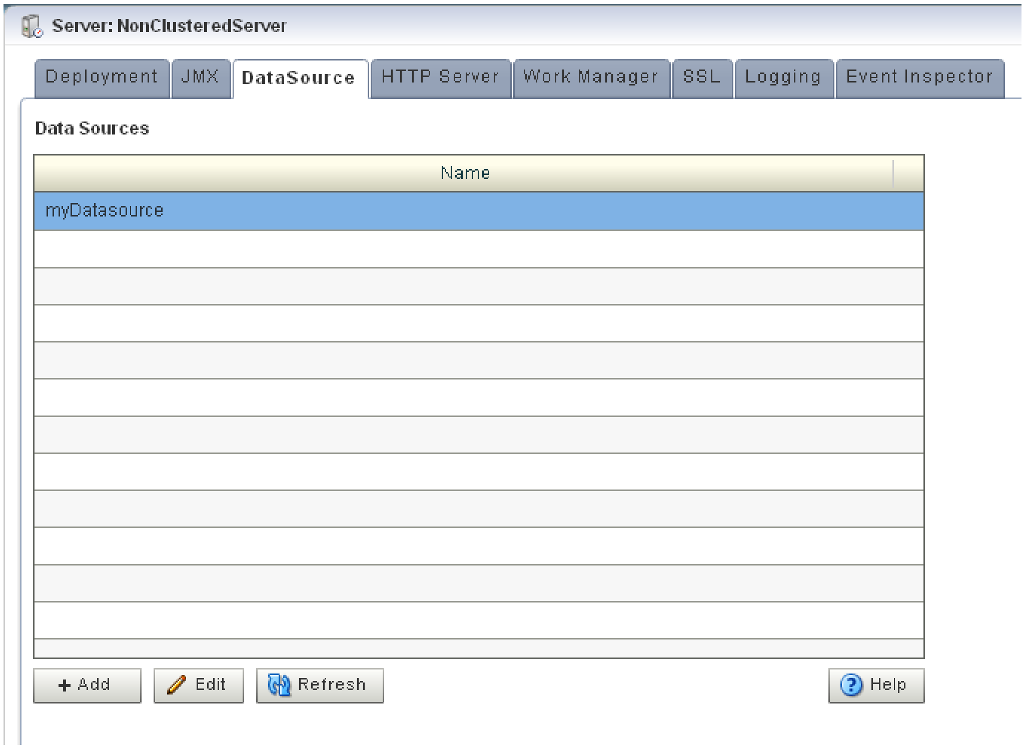
Using the Oracle Event Processing Visualizer, you can view the data source configuration for a selected Oracle Event Processing server.

View a Data Source Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 12-1 shows.

Figure 12-1 DataSource Tab



Server: NonClusteredServer

Data Sources

Name
myDatasource

12.2 Edit Data Source Configuration

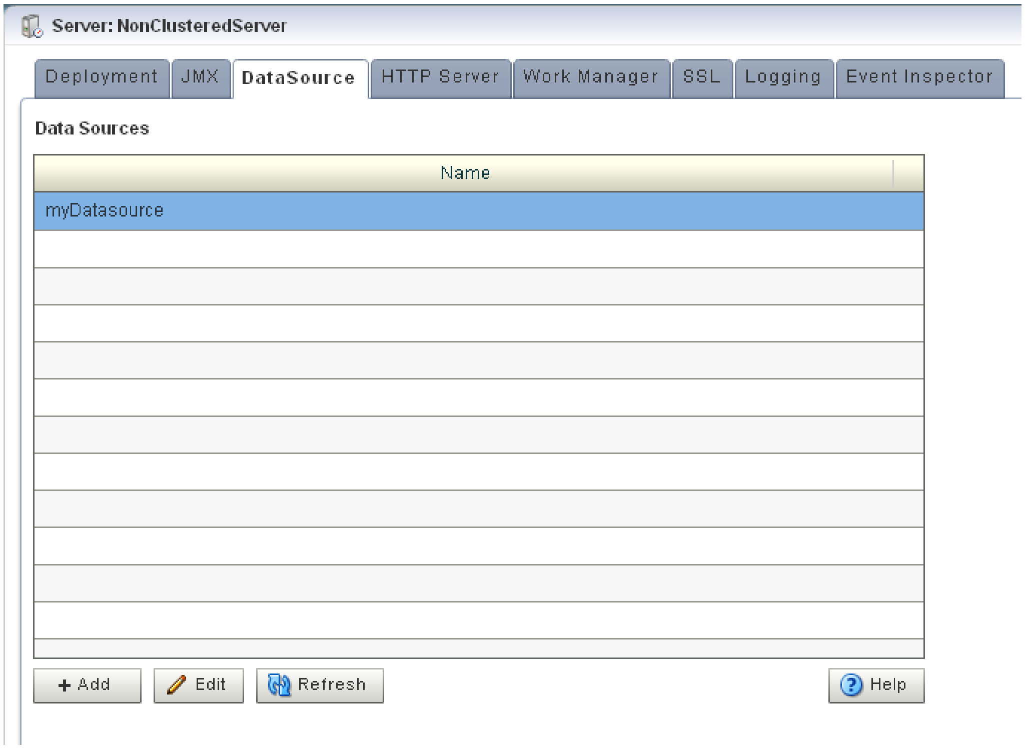
Using the Oracle Event Processing Visualizer, you can edit the data source configuration for a selected Oracle Event Processing server.

Edit Data Source Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 12-2 shows.

Figure 12-2 DataSource Tab



Name
myDatasource

3. In the Data Sources table, select the data source you want to edit by clicking on its name.

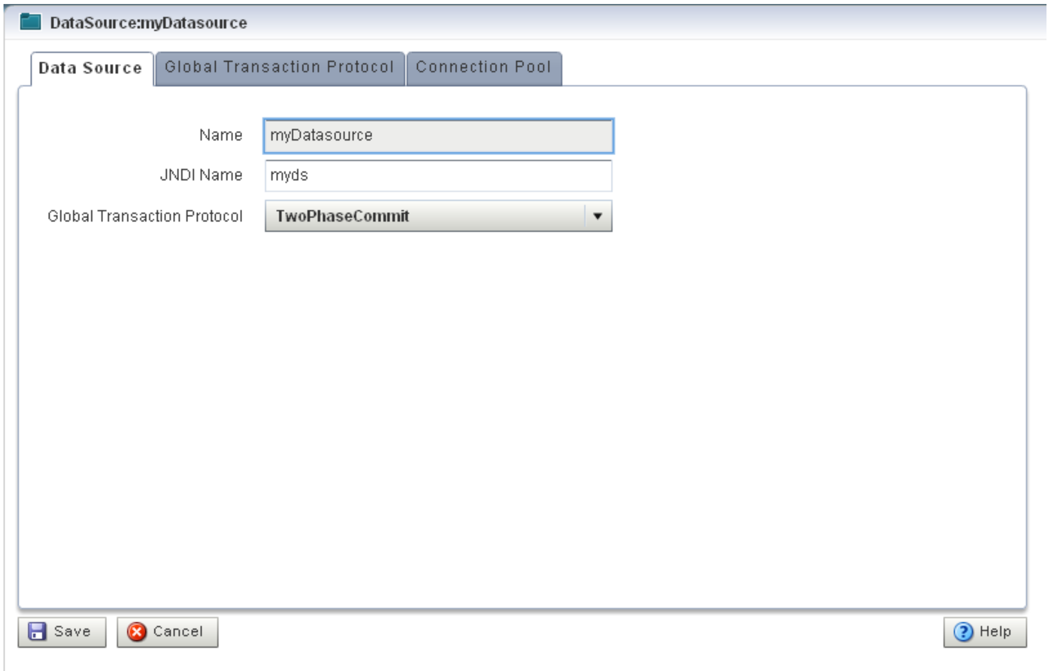
4. Click **Edit**.

   The Datasource dialog displays.

5. In the Datasource dialog, click the **Data Source** tab.

   The DataSource tab appears as Figure 12-3 shows.

Figure 12-3 Edit Datasource: Data Source Tab



DataSource:myDatasource

Data Source | Global Transaction Protocol | Connection Pool

Name: myDatasource

JNDI Name: myds

Global Transaction Protocol: TwoPhaseCommit

6. Configure the DataSource tab as Table 12-1 describes.

Table 12-1 Edit Datasource: Data Source Parameters

Parameter	Description
`Name`	The name for this data source configuration.
`JNDI Name`	JNDI path to where this data source will be bound. This is the name your Oracle Event Processing application uses to look up the data source in JNDI.
`Global Tx Protocol`	Select the protocol for Oracle Event Processing to use for the transaction branch when processing a global transaction: • `OnePhaseCommit` (Default) • `TwoPhaseCommit` • `LoggingLastResource` • `EmulateTwoPhaseCommit` • `None`

7. Click the **Global Tx Protocol** tab.

   The Global Tx Protocol tab displays.

8. Configure the Global Tx Protocol tab as Table 12-2 describes.

Table 12-2 Edit Datasource: Global Tx Protocol Parameters

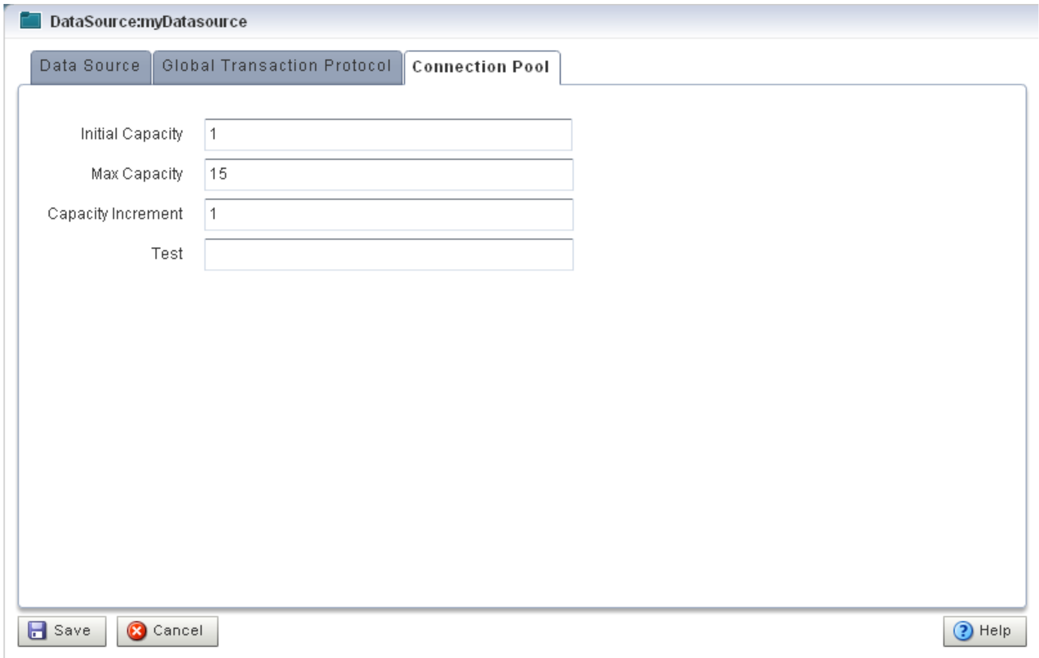
Parameter	Description
`Database Type`	Select the type of JDBC driver to use: • `Oracle` • `SQL Server 2005` • `Other`

Parameter	Description
`URL`	Specify the database connection URL appropriate for your database. This URL includes the name of the database, the host and port of the computer on which the database server is running, and so on. Oracle Event Processing populates a default URL for database types it provides.
`Driver Name`	Specify the name of the driver for the selected database type. Oracle Event Processing automatically populates the driver name for types it provides.
`User Name`	Specify database user account name that you want to use for each connection in the data source.
`Password`	Specify database user password that you want to use for each connection in the data source
`Use XA`	Specify whether or not to use distributed transactions: • `true`: enable distributed transactions. • `false`: disable distributed transactions.

9. Click the **Connection Pool** tab.

   The Connection Pool tab appears as Figure 12-4 shows.

Figure 12-4 Edit Datasource: Connection Pool Tab



DataSource:myDatasource

Initial Capacity: 1

Max Capacity: 15

Capacity Increment: 1

Test:

10. Configure the Connection Pool tab as Table 12-3 describes.

Table 12-3 Edit Datasource: Connection Pool Parameters

Parameter	Description
`Initial Capacity`	Specify the number of physical connections to create when creating the connection pool.

Parameter	Description
`Max Capacity`	Specify the maximum number of physical connections that this connection pool can contain.
`Capacity Increment`	Specify how many connections to add when all the connections in the connection pool are in use and the connection pool receives a new connection request.
`Test`	Enter the name of a small table to use in a query if you want to test the database connection before handing it to the requesting application. The standard query is `select 1 from` *`table_name`*. If you prefer to use a different query as a connection test, enter `SQL` followed by a space and the SQL code you want to use to test database connections.

11. Click **Save**.

    The datasource is updated in the Oracle Event Processing server configuration.

12.3 Add a Data Source Configuration

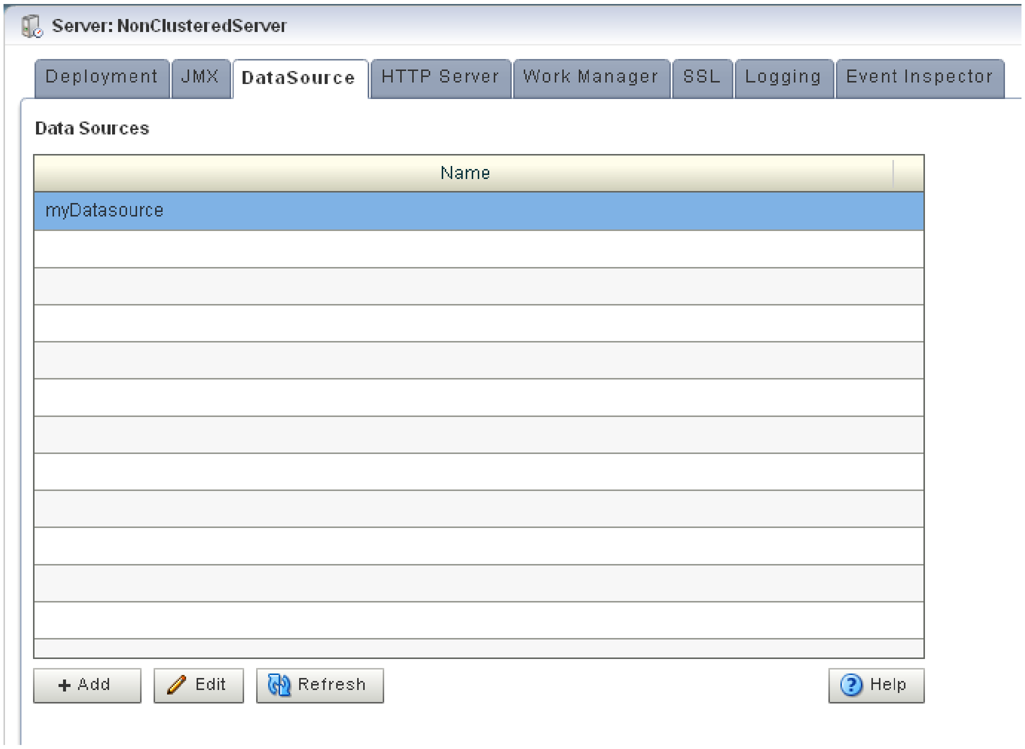
Using the Oracle Event Processing Visualizer, you can add the data source configuration for a selected Oracle Event Processing server.

Add a Data Source Configuration

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **DataSource** tab.

   The DataSource tab appears as Figure 12-5 shows.

Figure 12-5 DataSource Tab



Name
myDatasource

3. Click **Add**.

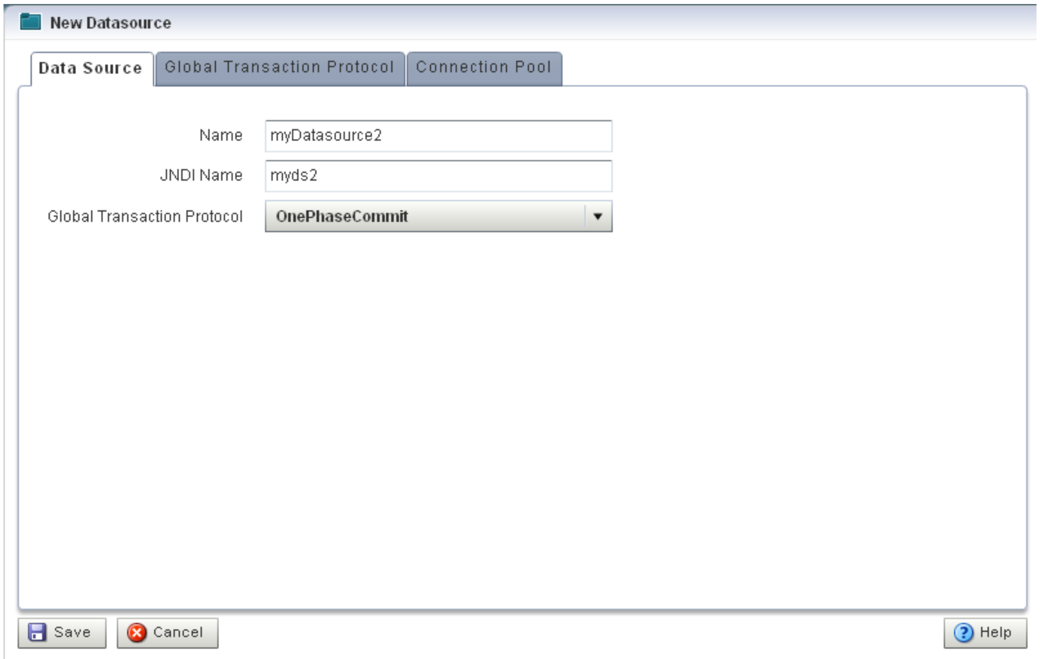
   The New Datasource dialog appears as Figure 12-6 shows. There are three accordion tabs:

   - **Data Source**: use this tab to identify the data source and specify whether or not to use a global transaction protocol.
   - **Global Tx Protocol**: use this tab to specify connection properties and whether or not to use XA.
   - **Connection Pool**: use this tab to specify connection pool properties.

4. Click the **Data Source** tab.

   The DataSource tab appears as Figure 12-6 shows.

Figure 12-6 New Datasource: Data Source Tab



Name: myDatasource2

JNDI Name: myds2

Global Transaction Protocol: OnePhaseCommit

5. Configure the DataSource tab as Table 12-4 describes.

Table 12-4 New Datasource: Data Source Parameters

Parameter	Description
`Name`	The name for this data source configuration.
`JNDI Name`	JNDI path to where this data source will be bound. This is the name your Oracle Event Processing application uses to look up the data source in JNDI.
`Global Tx Protocol`	Select the protocol for Oracle Event Processing to use for the transaction branch when processing a global transaction: • `OnePhaseCommit` (Default) • `TwoPhaseCommit` • `LoggingLastResource` • `EmulateTwoPhaseCommit` • `None`

6. Click the **Global Tx Protocol** tab.

   The Global Tx Protocol tab displays.

7. In the Global Tx Protocol tab, set the configuration as Table 12-5 describes.

Table 12-5 New Datasource: Global Tx Protocol Parameters

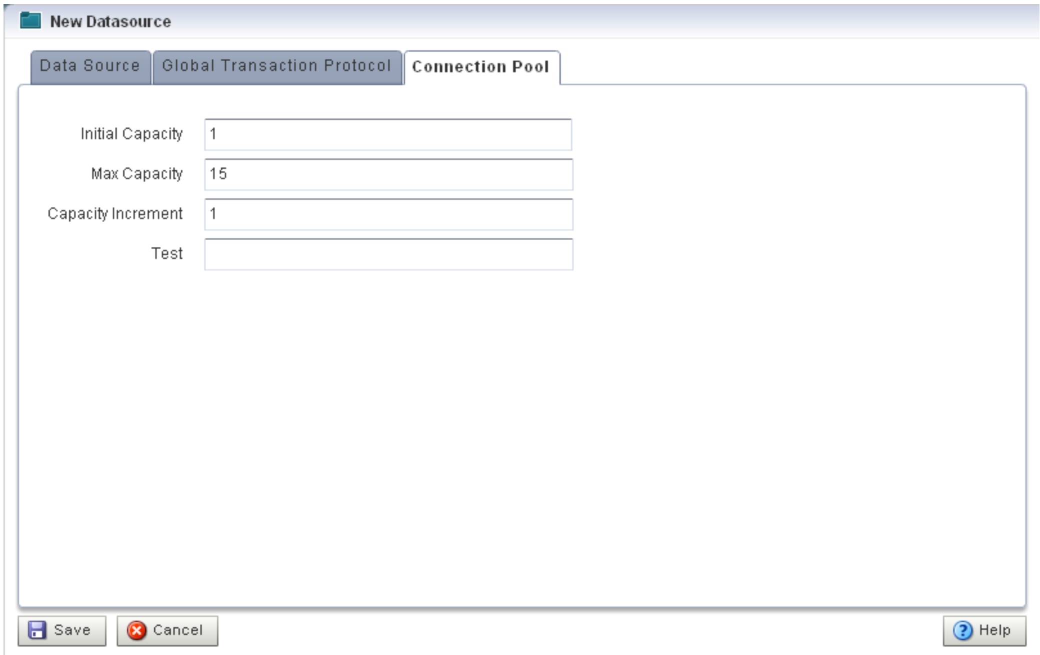
Parameter	Description
`Database Type`	Select the type of JDBC driver to use: • `Oracle` • `SQL Server 2005` • `Other`

Parameter	Description
`URL`	Specify the database connection URL appropriate for your database. This URL includes the name of the database, the host and port of the computer on which the database server is running, and so on. Oracle Event Processing populates a default URL for database types it provides.
`Driver Name`	Specify the name of the driver for the selected database type. Oracle Event Processing automatically populates the driver name for types it provides.
`User Name`	Specify database user account name that you want to use for each connection in the data source.
`Password`	Specify database user password that you want to use for each connection in the data source
`Use XA`	Specify whether or not to use distributed transactions: • `true`: enable distributed transactions. • `false`: disable distributed transactions.

8. Click the **Connection Pool** tab.

   The Connection Pool tab appears as Figure 12-7 shows.

Figure 12-7 New Datasource: Connection Pool Tab



New Datasource

Initial Capacity: 1

Max Capacity: 15

Capacity Increment: 1

Test:

9. Configure the Connection Pool tab as Table 12-6 describes.

Table 12-6 New Datasource: Connection Pool Parameters

Parameter	Description
`Initial Capacity`	Specify the number of physical connections to create when creating the connection pool.

Parameter	Description
`Max Capacity`	Specify the maximum number of physical connections that this connection pool can contain.
`Capacity Increment`	Specify how many connections to add when all the connections in the connection pool are in use and the connection pool receives a new connection request.
`Test`	Enter the name of a small table to use in a query if you want to test the database connection before handing it to the requesting application. The standard query is `select 1 from` *`table_name`*. If you prefer to use a different query as a connection test, enter `SQL` followed by a space and the SQL code you want to use to test database connections.

10. Click **Save**.

    The new datasource is added to the Oracle Event Processing server configuration.

13

HTTP Publish-Subscribe Servers

This chapter describes how you can use Oracle Event Processing Visualizer to view, add, or delete HTTP publish-subscribe server channels for use with Oracle Event Processing.

This chapter includes the following sections:

- View an HTTP Publish-Subscribe Server Channels
- Add an HTTP Publish-Subscribe Server Channel
- Delete an HTTP Publish-Subscribe Server Channel.

For more information, see:

- HTTP Publish-Subscribe Server Management
- Event Inspector Service

  HTTP Publish-Subscribe Server Security.

13.1 View an HTTP Publish-Subscribe Server Channels

Using the Oracle Event Processing Visualizer, you can view the HTTP publish-subscribe server channels currently open on the Oracle Event Processing server, including:

- **Static:** those channels configured in the Oracle Event Processing server `config.xml` and used by Oracle Event Processing Visualizer itself.
- **Dynamic:** those channels that are not configured in the Oracle Event Processing server `config.xml` but created dynamically when the Oracle Event Processing sever is running. You can view statistics for dynamic channels, including the number of messages sent to this channel and the number of subscribers. Dynamic channel configuration is read-only.

View HTTP Publish-Subscribe Server Channels

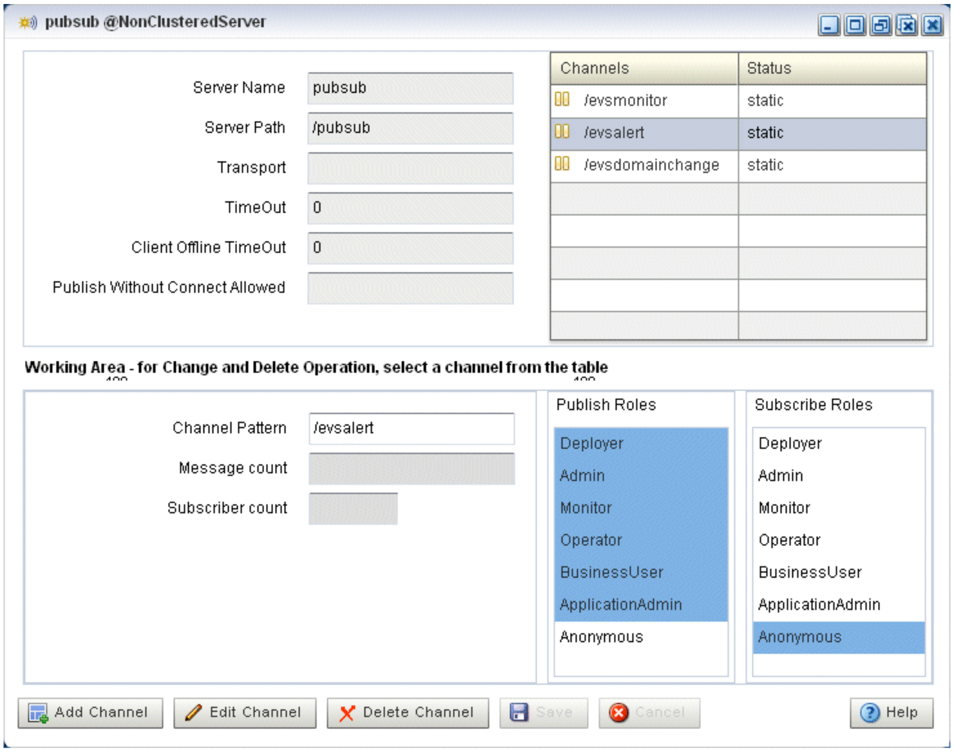
1. Be sure you log on to Oracle Event Processing Visualizer using an administration user.
2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub Server** node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle Event Processing.

3. In the right pane, double-click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.

   The HTTP publish-subscribe server properties dialog appears as Figure 13-1 shows.

Figure 13-1 HTTP Publish-Subscribe Server Channel List



pubsub @NonClusteredServer

Server Name	pubsub
Server Path	/pubsub
Transport	
TimeOut	0
Client Offline TimeOut	0
Publish Without Connect Allowed	

Channels	Status
/evsmonitor	static
/evsalert	static
/evsdomainchange	static

Working Area - for Change and Delete Operation, select a channel from the table

Channel Pattern	/evsalert
Message count	
Subscriber count	

Publish Roles	Subscribe Roles
Deployer	Deployer
Admin	Admin
Monitor	Monitor
Operator	Operator
BusinessUser	BusinessUser
ApplicationAdmin	ApplicationAdmin
Anonymous	Anonymous

All active channels are shown in the Channels table. The **Status** column indicates the type of channel.

13.2 Add an HTTP Publish-Subscribe Server Channel

Using the Oracle Event Processing Visualizer, you can add an HTTP publish-subscribe server channel.

Add an HTTP Publish-Subscribe Server Channel

1. Be sure you log on to Oracle Event Processing Visualizer using an administration user.
2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub Server** node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle Event Processing.
3. In the right pane, double-click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.
4. Click **Add Channel**.

5. In the working area towards the bottom of the right pane, enter the name of the new channel in the **Channel Pattern** text box.

   The channel must start with a `/`, such as `/mychannel`.

6. If you have security enabled, select the security roles that are allowed to publish and subscribe to the channel.

   Select more than one by holding down the **Ctrl** key.

7. Click **Save**.

   The new channel appears in the Channels table at the top right of the pane.

13.3 Delete an HTTP Publish-Subscribe Server Channel

Using the Oracle Event Processing Visualizer, you can add an HTTP publish-subscribe server channel.

Delete an HTTP Publish-Subscribe Server Channel

1. Be sure you log on to Oracle Event Processing Visualizer using an administration user.

2. In the left pane, click the *Domain* > *Server* > **Services** > **Http Pub/Sub server** node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle Event Processing.

3. In the right pane, click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.

4. In the Channels table, click the name of the channel you want to delete.

5. Click **Delete Channel** at the bottom of the pane.

14

Event Inspector Service

You can use Oracle Event Processing Visualizer to configure the HTTP pub-sub server that the event inspector service uses on a selected Oracle Event Processing server.

For more information, see Event Inspector Service Management.

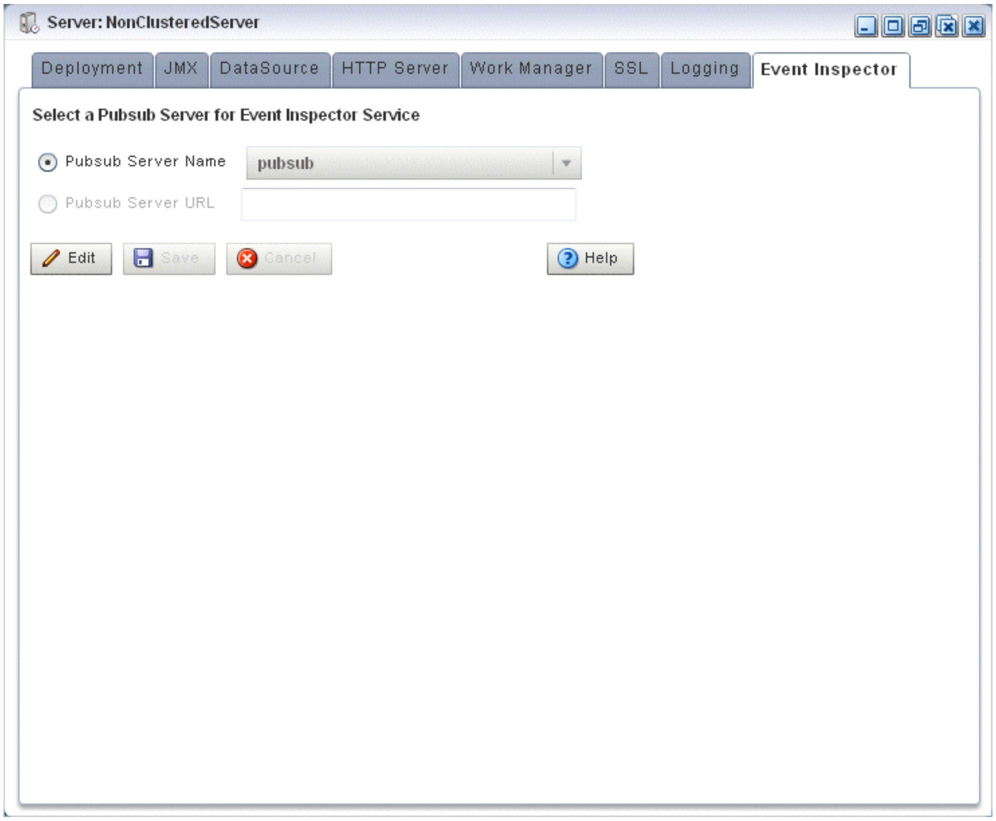
Configure the Event Inspector Service HTTP Publish-Subscribe Server

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. In the right pane, click the **Event Inspector** tab.

   The Event Inspector tab appears as Figure 14-1 shows.

   Figure 14-1 Event Inspector Tab



Server: NonClusteredServer

Select a Pubsub Server for Event Inspector Service

● Pubsub Server Name: pubsub

○ Pubsub Server URL:

3. Click **Edit**.

4. Edit the dialog as Table 14-1 describes.

Table 14-1 Event Inspector Options

Attribute	Description
`Pubsub Server Name`	Specify the name of the local HTTP pub-sub server that the Event Inspector service will use.
`Pubsub Server URL`	Specify the URL of the remote HTTP pub-sub server that the Event Inspector service will use. The URL should be in the form: `http://`*`HOST`*`:`*`PORT`*`/`*`PATH`* Where: • *`HOST`*: is the host name or IP address of the remote Oracle Event Processing server. • *`PORT`*: the remote Oracle Event Processing server `netio` port as defined in the remote Oracle Event Processing server `config.xml` file. Default: 9002. • *`PATH`*: the value of the `http-pubsub` element `path` child element as defined in the remote Oracle Event Processing server `config.xml` file. For example: `http://remotehost:9002/pubsub`

5. Click the **Save** button to commit your changes.

15

Event Type Repository

This chapter describes how you can use Oracle Event Processing Visualizer to view properties for an event available through the event repository.

This chapter includes the following sections:

- View the Event Type Repository
- Configure the Event Type Repository

For more information, see Event Type Repository Management.

15.1 View the Event Type Repository

Using the Oracle Event Processing Visualizer, you can view the event type repository.

View the Event Type Repository

1. In the left pane, click *Domain* > *Server* > **Services** > **Event Type Repository**, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

   The right pane displays the Event Name table that lists all the event types used by the applications of the server.

2. In the right pane, click the name of an event in the Event Name table. The text box at the bottom displays the properties of the event type. For example, a simple `Name` event type that has three properties, `first` and `last` which are `Strings` and `age` which is an integer, might look like:

   ```
   {first=java.lang.String, last=java.lang.String, age=int}
   ```

> **Note:**
>
> The default load generator and CSV adapters provide the `OracleL2StockTick`, `OracleStockTick`, and `OracleStockTrend` event types for you to use.

15.2 Configure the Event Type Repository

You can only view the event type repository, along with the its event types, configured for an Oracle Event Processing server using Oracle Event Processing Visualizer. See *Developing Applications for Oracle Event Processing*.

16

Persistent Event Store

This chapter describes how you can use Oracle Event Processing Visualizer to view properties of a persistent event store in use by Oracle Event Processing.

This chapter includes the following sections:

- View the Persistent Event Store
- Configure the Persistent Event Store

For more information, see Persistent Event Store Management.

16.1 View the Persistent Event Store

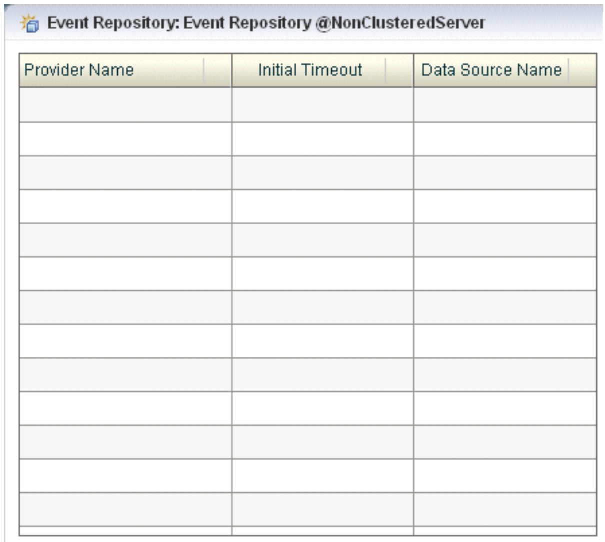
Using the Oracle Event Processing Visualizer, you can view the persistent event store.

View the Persistent Event Store

1. In the left pane, click *Domain* > *Server* > **Services** > **Event Repository**, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

   The Event Repository panel appears as Figure 16-1 shows. Table 16-1 describes the options in this panel.

   Figure 16-1 Event Repository Panel



Event Repository: Event Repository @NonClusteredServer

Provider Name	Initial Timeout	Data Source Name

Table 16-1 Event Repository Panel Attributes

Attribute	Description
Provider Name	The name of the persistent event store provider. Default: `default-provider`.
Initial Timeout	The data source timeout value.
Data Source Name	Displays the JDBC data source associated with the provider.

16.2 Configure the Persistent Event Store

You can only view the event store configured for an Oracle Event Processing server using Oracle Event Processing Visualizer. To change the configuration, you must manually update the server's `config.xml` file. See *Developing Applications for Oracle Event Processing*.

17

Multiserver Domains

This chapter describes how you can use Oracle Event Processing Visualizer to manage aspects of a multiserver domain (or cluster), including viewing native and Coherence cluster membership and topology diagrams, as well as cache information.

This chapter includes the following sections:

- View Cluster Group Membership
- View all Cluster Topologies
- View High Availability Cluster Topologies
- View Oracle Coherence Cluster Topologies
- Monitor Nodes in an Oracle Coherence Cluster
- Monitor an Oracle Coherence Cache
- Tune Oracle Coherence
- Manage the Cluster Topology Diagram.

17.1 View Cluster Group Membership

You can view the default and user-defined cluster groups, what servers belong to those groups, and what applications are deployed to those groups.

By default, there is a group for each server with the same name as the server and a group named AllDomainMembers which represents all the servers in the domain. In addition, you can define custom groups when you configure your cluster.

You may deploy an application to a cluster group or to an individual server. When you deploy to a cluster group, the application belongs to the group, not the individual servers in the group. When you deploy to an individual server, the application belongs to the server's own group.

For more information, see

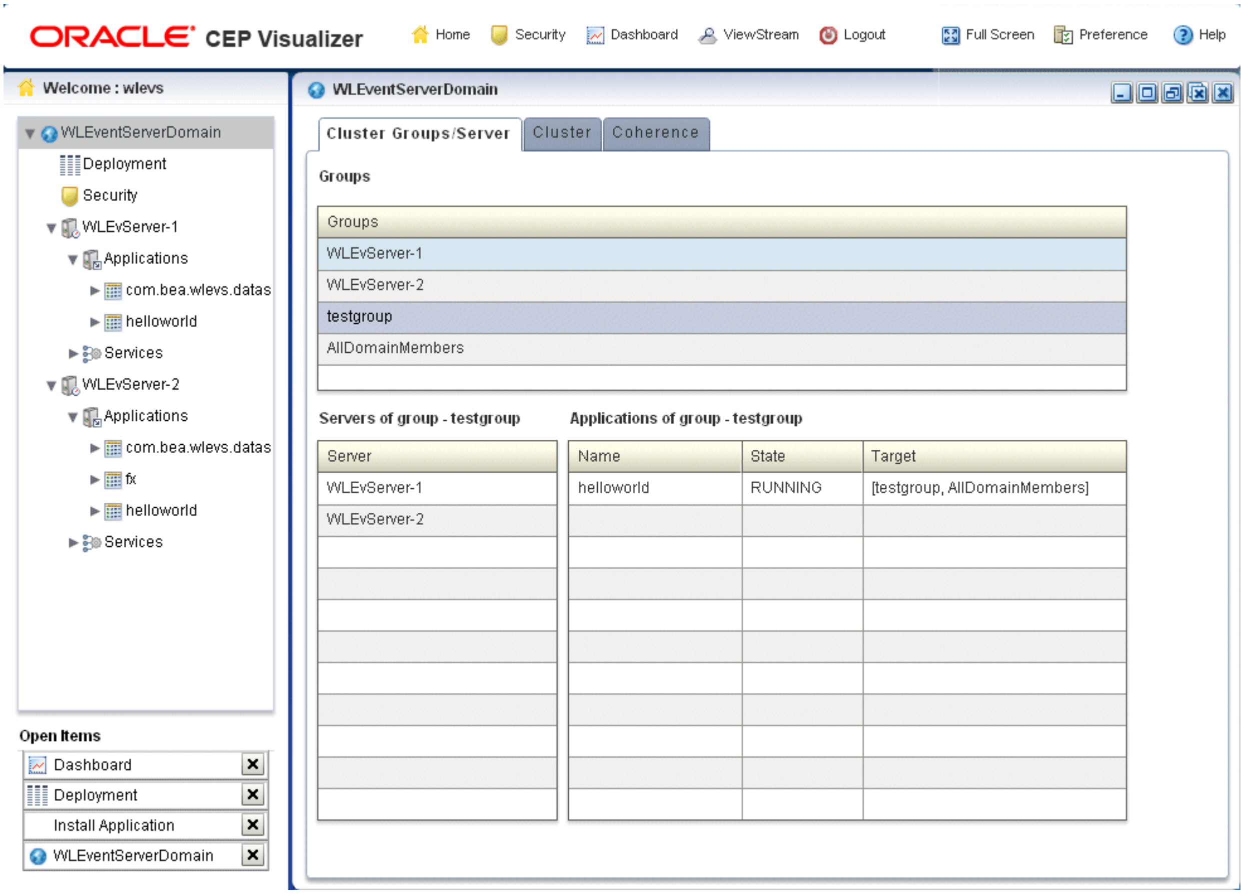
- Deploy an Application in a Multiserver Domain
- *Administering Oracle Event Processing*.

View Cluster Group Membership

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.
2. In the right pane, click the **Cluster Groups/Server** tab.

The Cluster Groups/Server tab appears as Figure 17-1 shows.

Figure 17-1 Cluster Groups/Server Tab



Groups

Groups
WLEvServer-1
WLEvServer-2
testgroup
AllDomainMembers

Servers of group - testgroup

Server
WLEvServer-1
WLEvServer-2

Applications of group - testgroup

Name	State	Target
helloworld	RUNNING	[testgroup, AllDomainMembers]

3. Use the Cluster Groups/Server tab to examine the server and application contents of various cluster groups.

   For example:

   - Group `testgroup` contains two servers (`WLEvServer-1` and `WLEvServer-2`) and one application (`helloworld`) as Figure 17-1 shows.
   - Group `WLEvServer-1` contains one server (`WLEvServer-1`) and no applications. The `helloworld` application listed under its `Applications` node belongs to group `testgroup`, not to server `WLEvServer-1`.
   - Group `WLEvServer-2` contains one server (`WLEvServer-2`) and one application (`fx`). The `fx` application was deployed to server `WLEvServer-2`. The `helloworld` application listed under its `Applications` node belongs to group `testgroup`, not to server `WLEvServer-2`.

17.2 View all Cluster Topologies

Using Oracle Event Processing Visualizer, you can view all cluster topologies regardless of cluster implementation. You can also view cluster group properties and server instance properties.

If you deploy an Oracle Event Processing high availability application, you can view the Oracle Event Processing high availability topology. For more information, see View High Availability Cluster Topologies.

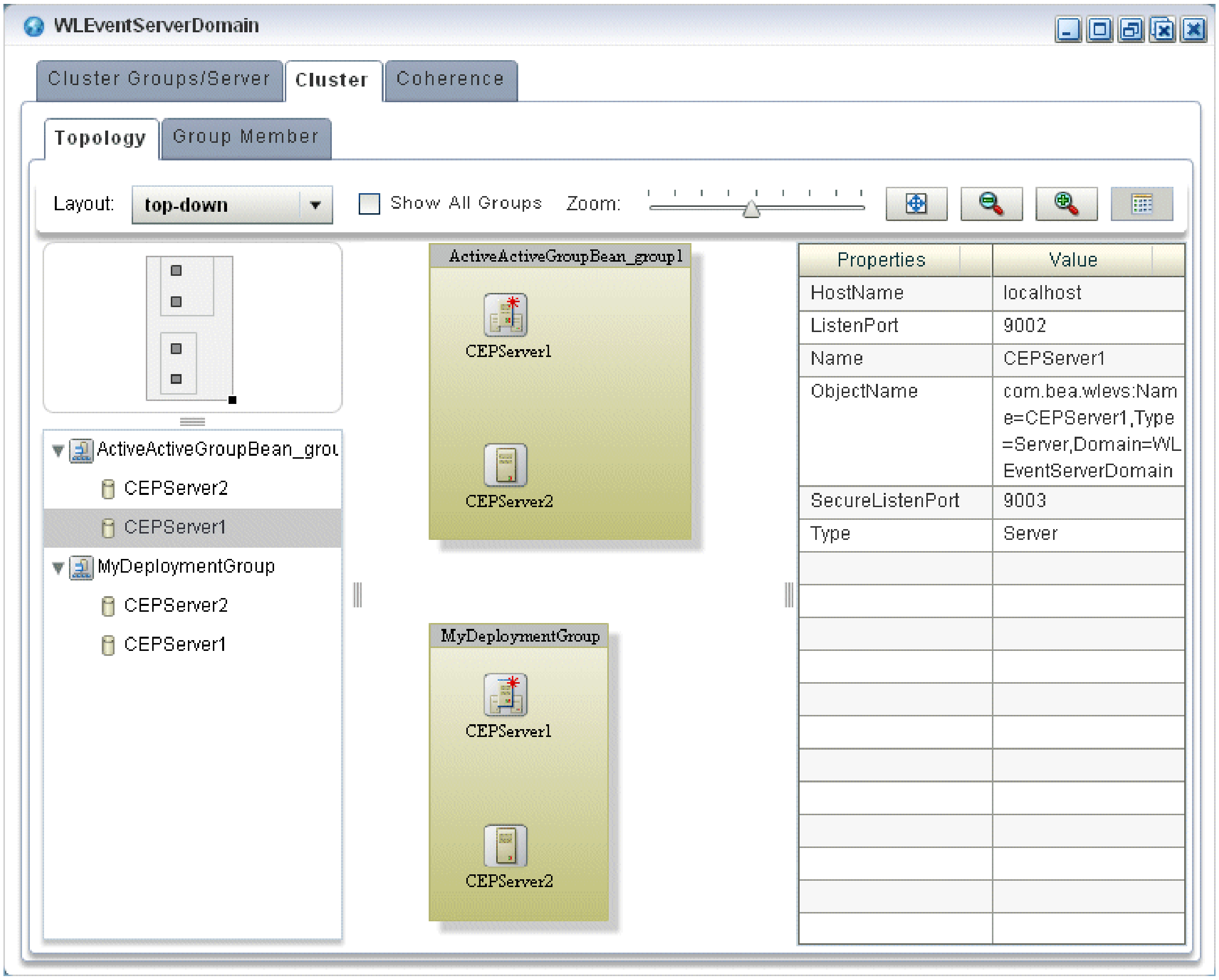
If you are using Oracle Coherence for clustering, you can view Oracle Coherence topology. For more information, see View Oracle Coherence Cluster Topologies.

View All Cluster Topologies

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.
2. In the right pane, click the **Cluster** tab.
3. In the right pane, click the **Topology** tab.

   The Topology tab appears as Figure 17-2 shows.

Figure 17-2 Cluster Topology Tab



Layout: top-down ☐ Show All Groups Zoom:

Properties	Value
HostName	localhost
ListenPort	9002
Name	CEPServer1
ObjectName	com.bea.wlevs:Name=CEPServer1,Type=Server,Domain=WLEventServerDomain
SecureListenPort	9003
Type	Server

This tab shows all cluster groups regardless of the cluster implementation.

Within a cluster group, the primary server is identified by the icon that Figure 17-3 shows. In Figure 17-2, the primary server is `CEPServer1`.

Figure 17-3 Primary Server

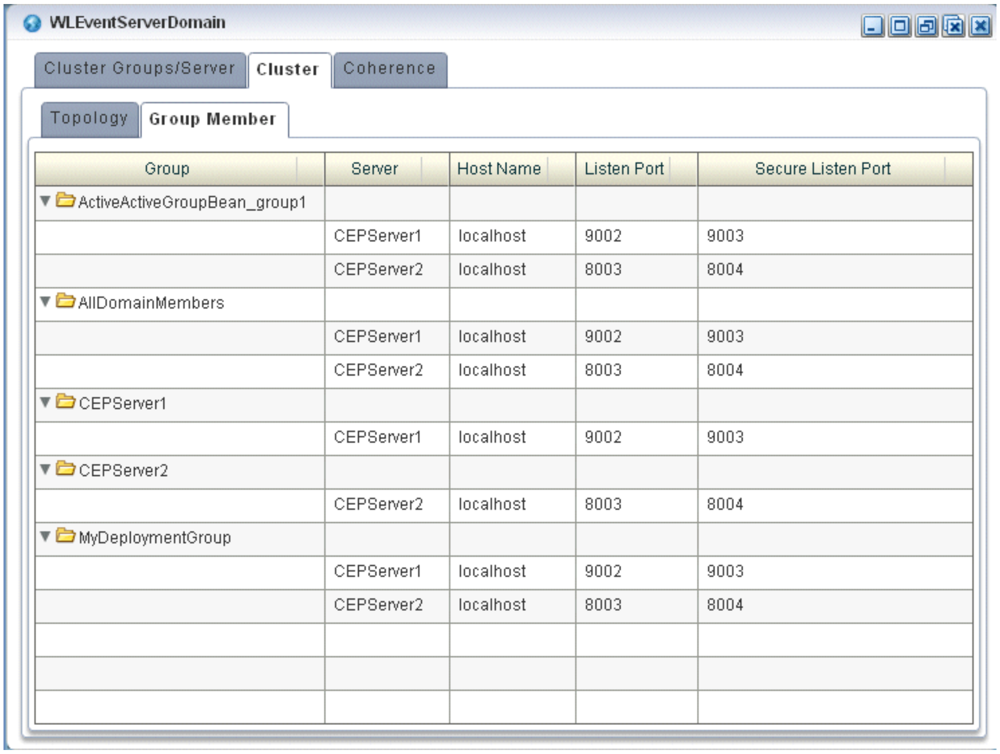
For information on the various tools along the top of the topology canvas, see Manage the Cluster Topology Diagram.

4. To view the properties for a given group, click on the group in the Group List.

5. To view the properties for a given server instance, click on the server instance in the Group List.

6. To view details on a given server instance, click the **Group Member** tab and select the server instance as Figure 17-4 shows.

Figure 17-4 Group Member Tab



Group	Server	Host Name	Listen Port	Secure Listen Port
▼ ActiveActiveGroupBean_group1				
	CEPServer1	localhost	9002	9003
	CEPServer2	localhost	8003	8004
▼ AllDomainMembers				
	CEPServer1	localhost	9002	9003
	CEPServer2	localhost	8003	8004
▼ CEPServer1				
	CEPServer1	localhost	9002	9003
▼ CEPServer2				
	CEPServer2	localhost	8003	8004
▼ MyDeploymentGroup				
	CEPServer1	localhost	9002	9003
	CEPServer2	localhost	8003	8004

7. Use the Group Member tab to customize the group table:

   - To change the order of columns, click on the column name in the cache chart and drag left or right.

   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

17.3 View High Availability Cluster Topologies

Using Oracle Event Processing Visualizer, you can view an Oracle high availability cluster topology. You can also view cluster group properties and server instance properties.

View High Availability Cluster Topologies

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.

2. In the right pane, click the **Cluster** tab.

3. In the right pane, click the **Topology** tab.

   For an Oracle Event Processing high availability application, you can use the Topology tab to:

   a. Visualize a high-availability configuration.

      See Visualize a High Availability Configuration

   b. Visualizer a high-availability and scalability configuration.

      See Visualize a High Availability and Scalability Configuration.

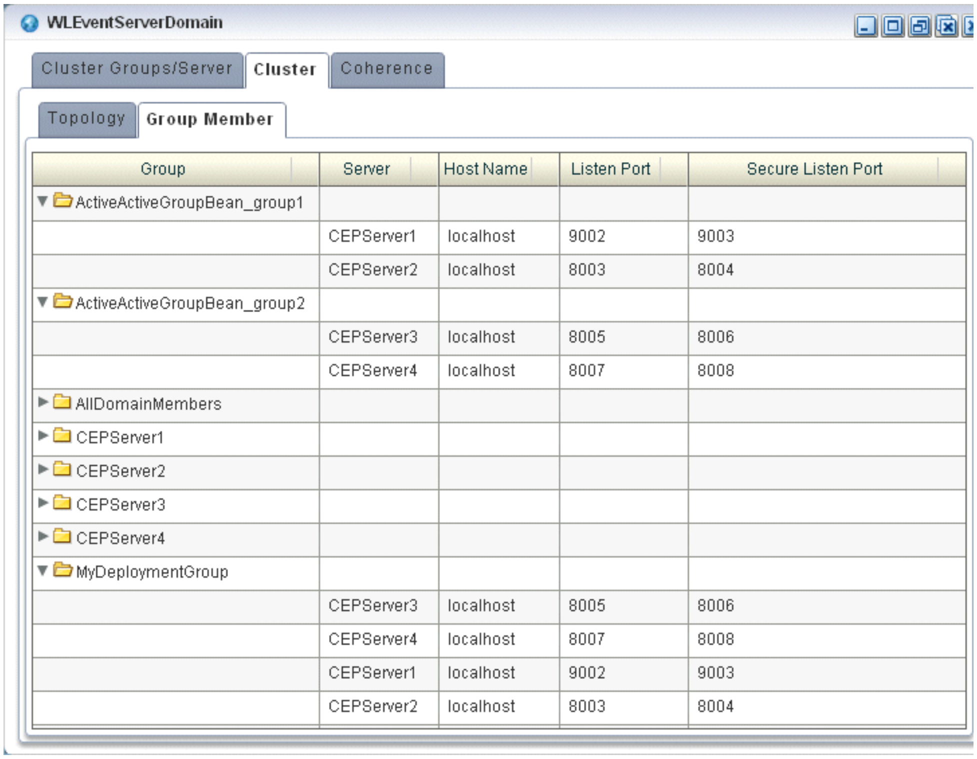
   For information on the various tools along the top of the topology canvas, see Manage the Cluster Topology Diagram.

4. To view the properties for a given group, click on the group in the Group List.

5. To view the properties for a given server instance, click on the server instance in the Group List.

6. To view details on a given server instance, click the **Group Member** tab and select the group and server instance as Figure 17-5 shows.

Figure 17-5 Group Member Tab



Group	Server	Host Name	Listen Port	Secure Listen Port
▼ ActiveActiveGroupBean_group1				
	CEPServer1	localhost	9002	9003
	CEPServer2	localhost	8003	8004
▼ ActiveActiveGroupBean_group2				
	CEPServer3	localhost	8005	8006
	CEPServer4	localhost	8007	8008
▶ AllDomainMembers				
▶ CEPServer1				
▶ CEPServer2				
▶ CEPServer3				
▶ CEPServer4				
▼ MyDeploymentGroup				
	CEPServer3	localhost	8005	8006
	CEPServer4	localhost	8007	8008
	CEPServer1	localhost	9002	9003
	CEPServer2	localhost	8003	8004

7. Use the Group Member tab to customize the group table:

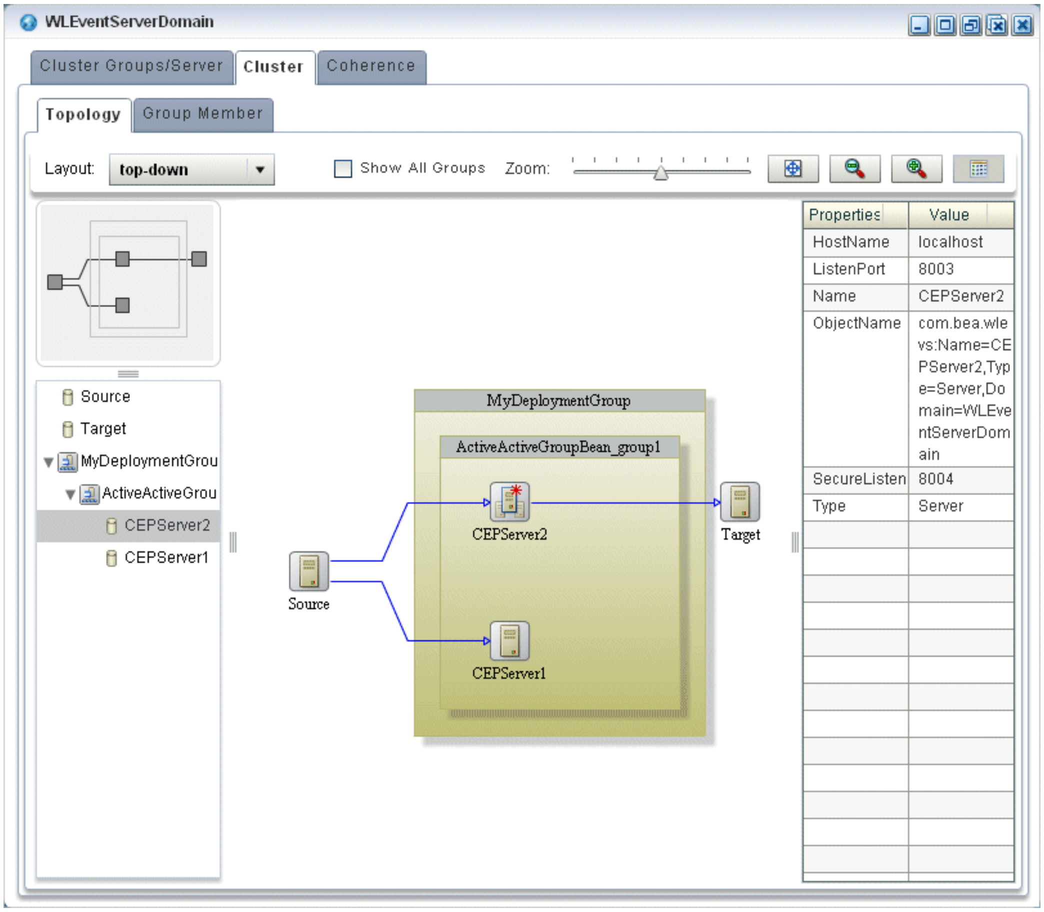
- To change the order of columns, click on the column name in the cache chart and drag left or right.
- To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

17.3.1 Visualize a High Availability Configuration

For an Oracle Event Processing high availability application, the Topology tab shows the notification group nested within the deployment group.

In Figure 17-6, the deployment group is `MyDeploymentGroup` and the notification group is `ActiveActiveGroupBean_group1`.

Figure 17-6 Cluster Topology Tab: Oracle Event Processing High Availability



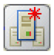
Layout: top-down

□ Show All Groups

Properties	Value
HostName	localhost
ListenPort	8003
Name	CEPServer2
ObjectName	com.bea.wlevs:Name=CEPServer2,Type=Server,Domain=WLEventServerDomain
SecureListen	8004
Type	Server

The primary server is identified by the icon that Figure 17-7 shows. In Figure 17-6, the primary server is `CEPServer2`.

Figure 17-7 Primary Server



The stream data source inputs to both primary and secondary servers but only the primary server outputs events. If `CEPServer2` goes down, an Oracle high availability failover occurs and `CEPServer1` resumes outputting events as the new primary for `ActiveActiveGroupBean_group1`. When `CEPServer2` is brought back online, it rejoins `ActiveActiveGroupBean_group1` as a secondary server.

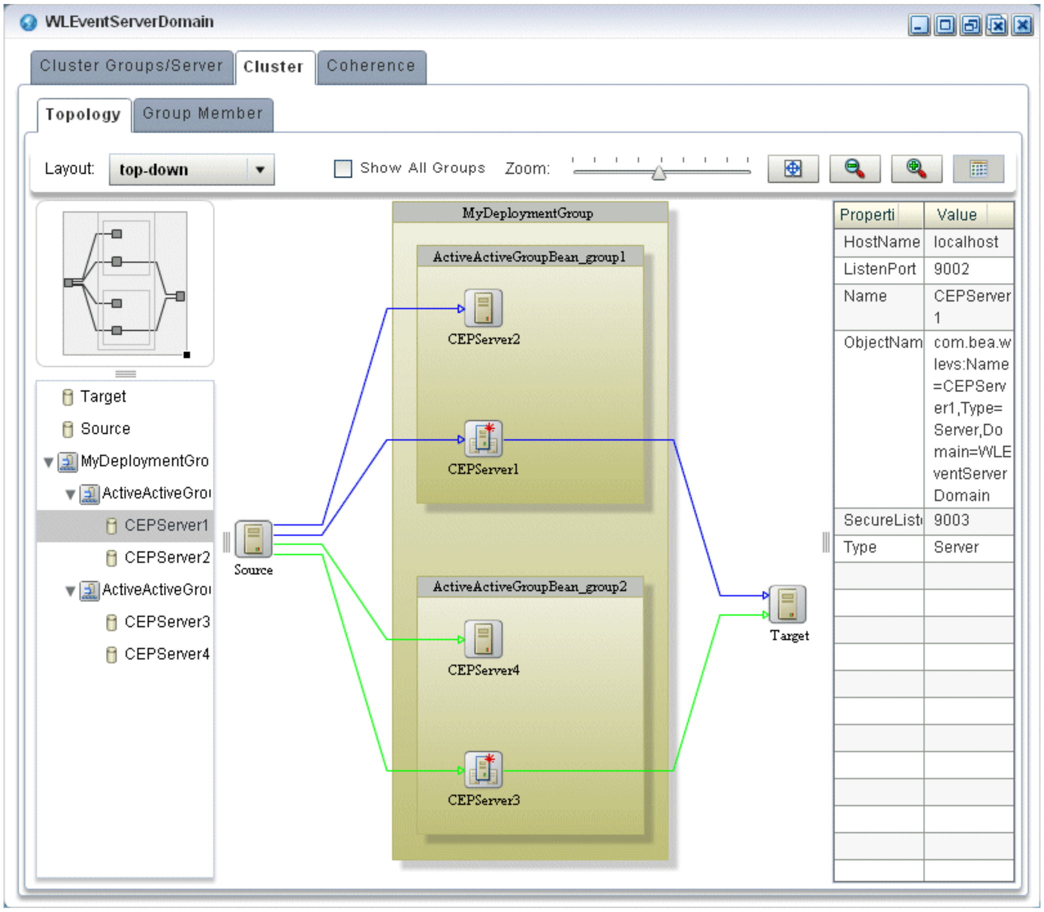
For information on the various tools along the top of the topology canvas, see Manage the Cluster Topology Diagram.

17.3.2 Visualize a High Availability and Scalability Configuration

If your Oracle Event Processing high availability application combines high availability and scalability, then the Topology tab shows notification groups nested within the deployment group.

In Figure 17-8, the deployment group is `MyDeploymentGroup` and the notification groups are `ActiveActiveGroupBean_group1` and `ActiveActiveGroupBean_group2`.

Figure 17-8 Cluster Topology Tab: Oracle Event Processing High Availability and Scalability



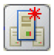
Layout: top-down

☐ Show All Groups

Properti	Value
HostName	localhost
ListenPort	9002
Name	CEPServer1
ObjectNam	com.bea.wlevs:Name=CEPServer1,Type=Server,Domain=WLEventServerDomain
SecureListe	9003
Type	Server

The primary server is identified by the icon that Figure 17-9 shows. In Figure 17-8, the primary server of notification group `ActiveActiveGroupBean_group1` is `CEPServer1` and the primary server of notification group `ActiveActiveGroupBean_group2` is `CEPServer3`.

Figure 17-9 Primary Server



The stream data source inputs to both primary and secondary servers in each notification group but only the primary server in each notification group outputs events. The stream data source is partitioned (by JMS selector) so that each notification group processes a different subset of the input data. The Oracle Event Processing Visualizer indicates this with a different colored line for each notification group. For example, `ActiveActiveGroupBean_group1` may process events with `accountID <= 500` and `ActiveActiveGroupBean_group2` may process events with `accountID > 500`. Both `CEPServer1` and `CEPServer2` process events with `accountID <= 500` in parallel and `CEPServer3` and `CEPServer4` process events with `accountID > 500` in parallel. If `CEPServer1` goes down, an Oracle high availability failover occurs and `CEPServer2` resumes outputting events as the new primary for `ActiveActiveGroupBean_group1`. When `CEPServer1` is brought back online, it rejoins `ActiveActiveGroupBean_group1` as a secondary server.

For information on the various tools along the top of the topology canvas, see Manage the Cluster Topology Diagram.

17.4 View Oracle Coherence Cluster Topologies

Using Oracle Event Processing Visualizer, if you are using Oracle Coherence clustering, you can view Oracle Coherence cluster topologies, cluster group properties, and server instance properties.

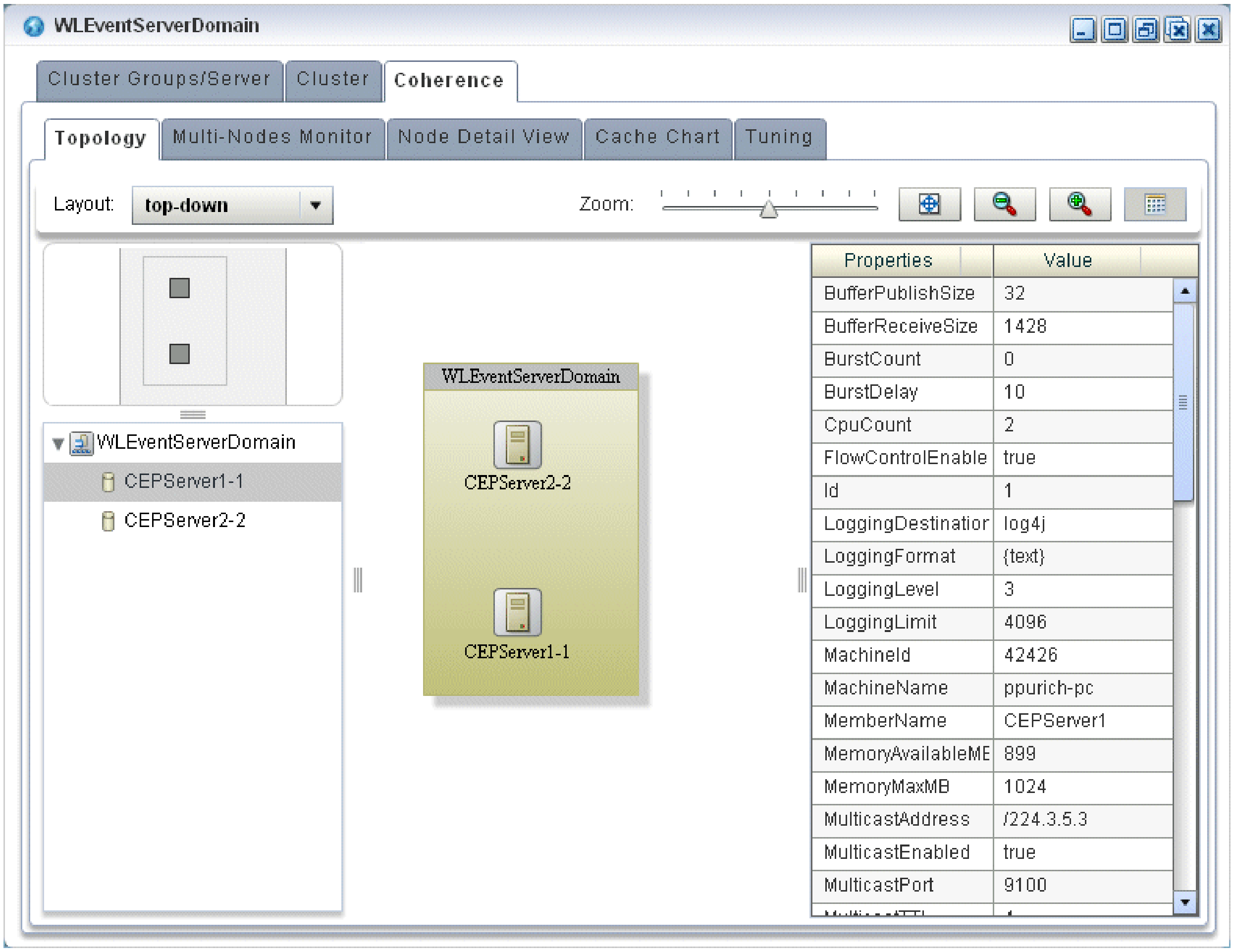
To view all clustering topologies regardless of clustering implementation, see View all Cluster Topologies.

View Oracle Coherence Cluster Topologies

1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.
2. In the right pane, click the **Coherence** tab.
3. In the right pane, click the **Topology** tab.

   The Topology tab appears as Figure 17-10 shows.

Figure 17-10 Oracle Coherence Cluster Topology Tab



Properties	Value
BufferPublishSize	32
BufferReceiveSize	1428
BurstCount	0
BurstDelay	10
CpuCount	2
FlowControlEnable	true
Id	1
LoggingDestinatior	log4j
LoggingFormat	{text}
LoggingLevel	3
LoggingLimit	4096
MachineId	42426
MachineName	ppurich-pc
MemberName	CEPServer1
MemoryAvailableME	899
MemoryMaxMB	1024
MulticastAddress	/224.3.5.3
MulticastEnabled	true
MulticastPort	9100

This tab shows all cluster groups that the Oracle Coherence clustering implementation manages.

For information on the various tools along the top of the topology canvas, see Manage the Cluster Topology Diagram.

4. To view the properties for a given group, click on the group in the Group List.

5. To view the properties for a given server instance, click on the server instance in the Group List.

17.5 Monitor Nodes in an Oracle Coherence Cluster

Using Oracle Event Processing Visualizer, you can compare a combination of properties for any and all nodes in an Oracle Coherence cluster. For a selected node in an Oracle Coherence cluster, you can view a select list of properties.

Once you have determined the root cause of a problem, you can tune Oracle Coherence parameters to correct the problem. For more information, see Tune Oracle Coherence.

Monitor Nodes in an Oracle Coherence Cluster

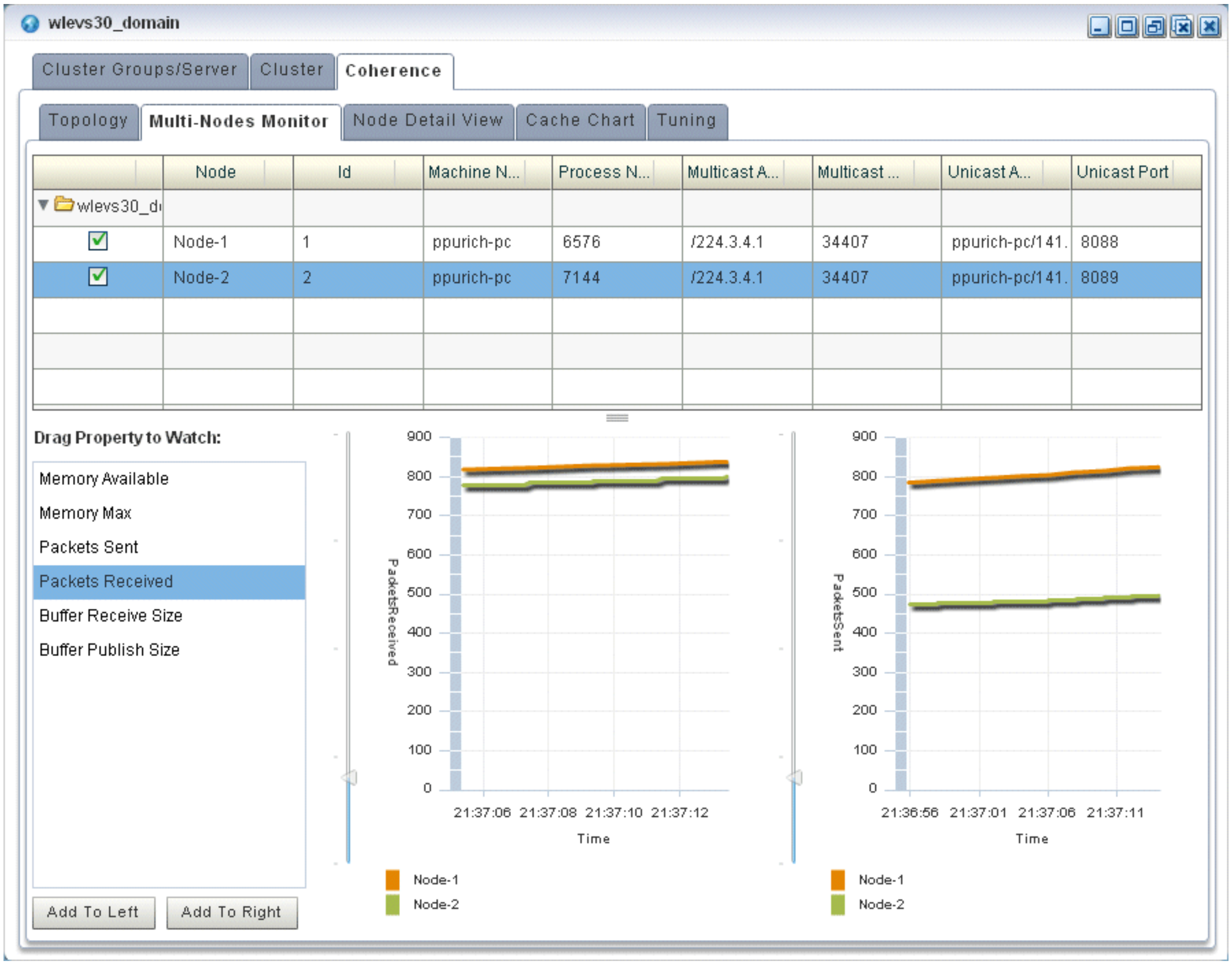
1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.

2. In the right pane, click the **Coherence** tab.

3. In the right pane, click the **Multi-Nodes Monitor** tab.

   The Multi-Nodes Monitor tab appears as Figure 17-11 shows.

Figure 17-11 Multi-Nodes Monitor Tab



	Node	Id	Machine N...	Process N...	Multicast A...	Multicast ...	Unicast A...	Unicast Port
▼ wlevs30_d								
☑	Node-1	1	ppurich-pc	6576	/224.3.4.1	34407	ppurich-pc/141.	8088
☑	Node-2	2	ppurich-pc	7144	/224.3.4.1	34407	ppurich-pc/141.	8089

4. Use the Multi-Nodes Monitor tab to customize the group table:

   - To change the order of columns, click on the column name in the cache chart and drag left or right.
   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

5. Use the Multi-Nodes Monitor tab to compare any combination of the properties that Table 17-1 lists for any nodes in an Oracle Coherence cluster:

   - Expand a group and check one or more nodes.
   - Drag a property from the **Drag Property to Watch** area to either graph to watch that property or, select a property and click **Add to Left** or **Add to Right**.

     You may specify at most one property per graph.

- A line is drawn for each selected node. The legend below the graph distinguishes graph data for each selected node by color.
- To watch a different property, drag a new property and drop it onto the graph or, select a new property and click **Add to Left** or **Add to Right**.
- To change the scale of a graph, drag the slider next to the vertical axis.

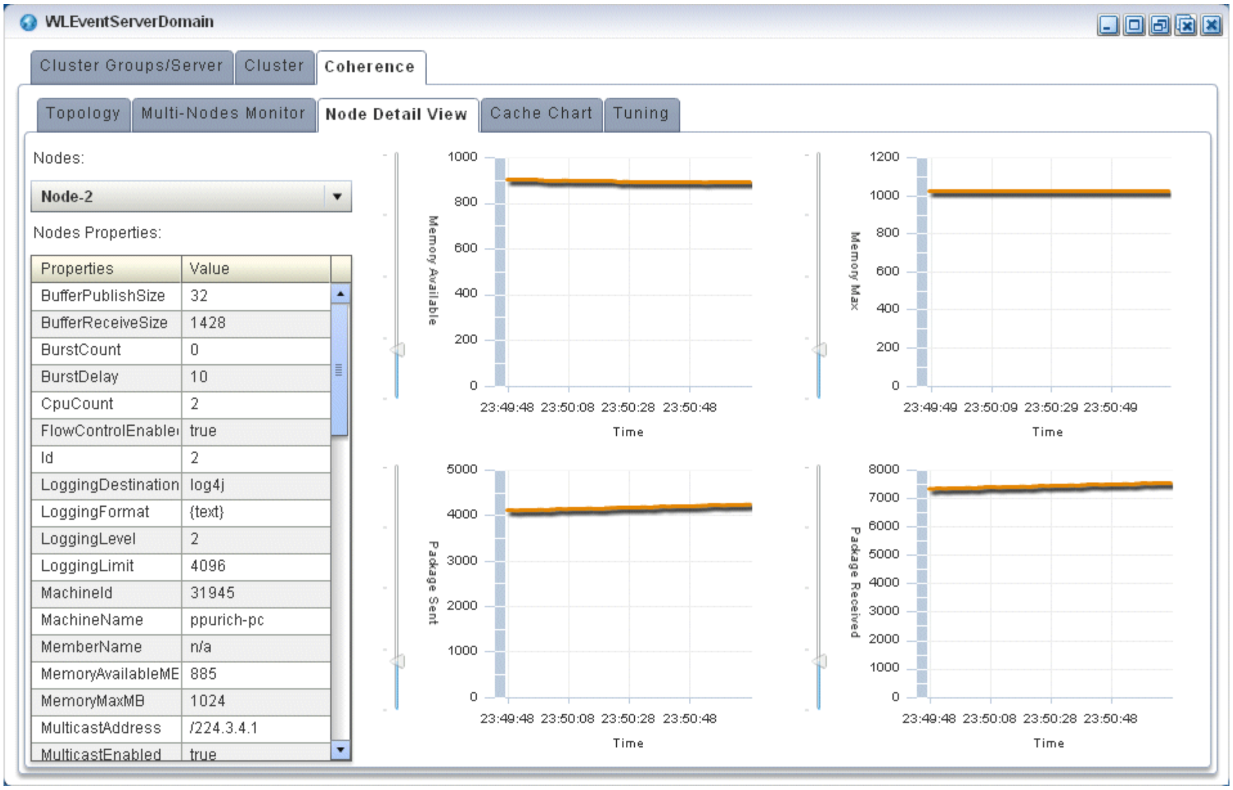
Table 17-1 Properties You Can Watch on the Multi-Nodes Monitor Tab

Property	Description
`Memory Available`	The total amount of free memory (in MB) for the selected node.
`Memory Max`	The total amount of memory available (in MB) for the selected node.
`Packets Sent`	The total number of Oracle Coherence packets sent by the selected node.
`Packets Received`	The total number of Oracle Coherence packets received by the selected node.
`Buffer Receive Size`	The total number of Oracle Coherence received packets that the selected node can buffer.
`Buffer Publish Size`	The total number of Oracle Coherence published packets that the selected node can buffer.

6. To view more details for a particular node, click the **Node Detail View** tab.

   The Node Detail View tab appears as Figure 17-12 shows.

Figure 17-12 Node Detail View Tab



Nodes: Node-2

Nodes Properties:

Properties	Value
BufferPublishSize	32
BufferReceiveSize	1428
BurstCount	0
BurstDelay	10
CpuCount	2
FlowControlEnable	true
Id	2
LoggingDestination	log4j
LoggingFormat	{text}
LoggingLevel	2
LoggingLimit	4096
MachineId	31945
MachineName	ppurich-pc
MemberName	n/a
MemoryAvailableME	885
MemoryMaxMB	1024
MulticastAddress	/224.3.4.1
MulticastEnabled	true

7. Use the Node Detail View tab to compare the properties that Table 17-2 lists for a selected node:

   - Select a node from the **Nodes** pull-down menu.

     The properties for the selected node are shown in the Nodes Properties list.

   - A line is drawn for the selected node.

   - To change the scale of a graph, drag the slider next to the vertical axis.

Table 17-2 Properties You Can Watch on the Node Detail View Tab

Property	Description
`Memory Available`	The total amount of free memory (in MB) for the selected node.
`Memory Max`	The total amount of memory available (in MB) for the selected node.
`Packets Sent`	The total number of Oracle Coherence packets sent by the selected node.
`Packets Received`	The total number of Oracle Coherence packets received by the selected node.

17.6 Monitor an Oracle Coherence Cache

Using Oracle Event Processing Visualizer, you can monitor a wide variety of Oracle Coherence cache parameters and graphically compare parameter values to locate and diagnose performance and operational problems.

Once you have determined the root cause of a problem, you can tune Oracle Coherence parameters to correct the problem. For more information, see Tune Oracle Coherence.

Monitor an Oracle Coherence Cache

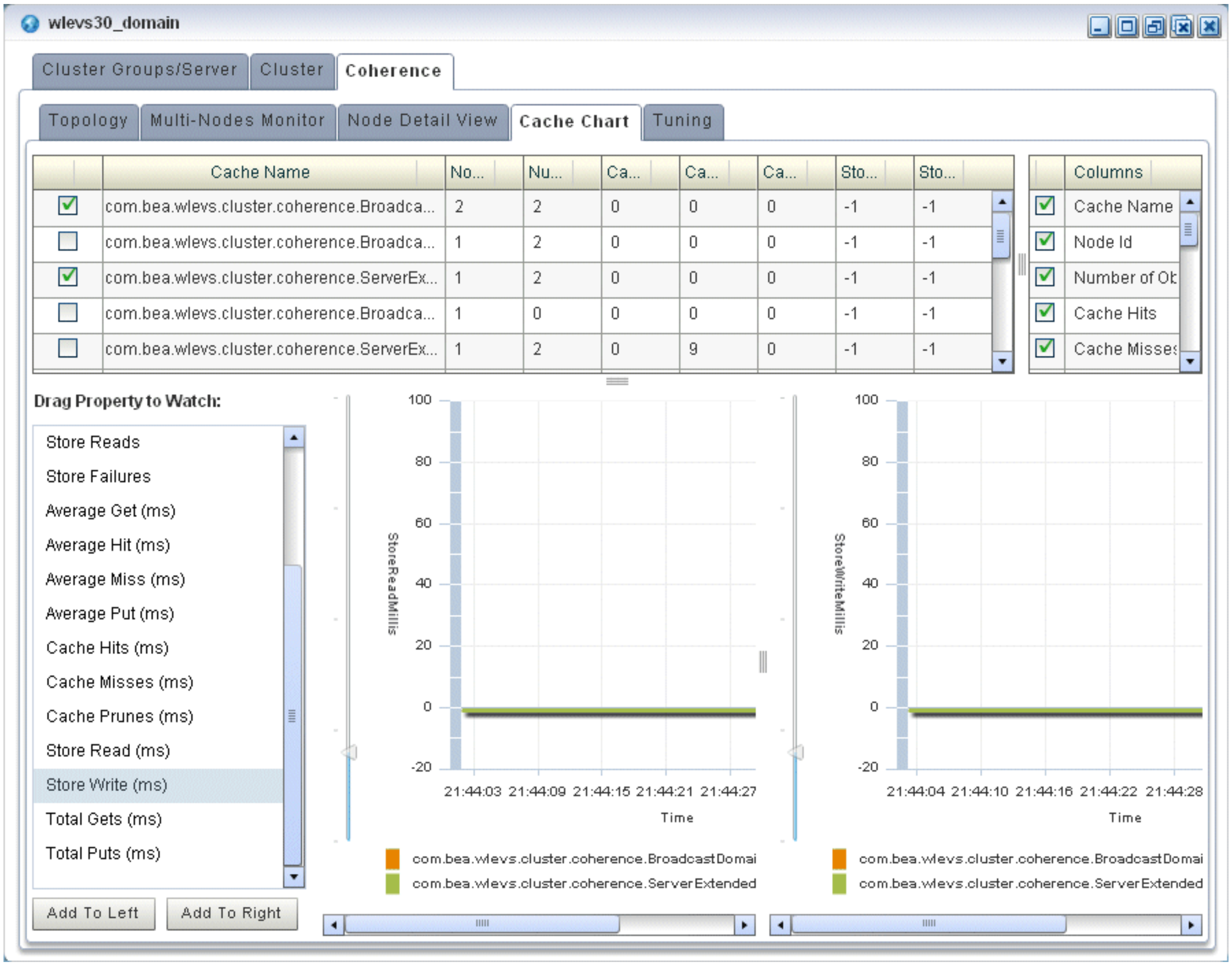
1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.

2. In the right pane, click the **Coherence** tab.

3. In the right pane, click the **Cache Chart** tab.

   The Multi-Nodes Monitor tab appears as Figure 17-13 shows.

Figure 17-13 Cache Chart Tab



	Cache Name	No...	Nu...	Ca...	Ca...	Ca...	Sto...	Sto...
☑	com.bea.wlevs.cluster.coherence.Broadca...	2	2	0	0	0	-1	-1
□	com.bea.wlevs.cluster.coherence.Broadca...	1	2	0	0	0	-1	-1
☑	com.bea.wlevs.cluster.coherence.ServerEx...	1	2	0	0	0	-1	-1
□	com.bea.wlevs.cluster.coherence.Broadca...	1	0	0	0	0	-1	-1
□	com.bea.wlevs.cluster.coherence.ServerEx...	1	2	0	9	0	-1	-1

	Columns
☑	Cache Name
☑	Node Id
☑	Number of Ob
☑	Cache Hits
☑	Cache Misses

Drag Property to Watch:

- Store Reads
- Store Failures
- Average Get (ms)
- Average Hit (ms)
- Average Miss (ms)
- Average Put (ms)
- Cache Hits (ms)
- Cache Misses (ms)
- Cache Prunes (ms)
- Store Read (ms)
- Store Write (ms)
- Total Gets (ms)
- Total Puts (ms)

Add To Left   Add To Right

4. Use the Cache Chart tab to view any combination of the columns that Table 17-3 lists for one or more selected caches:

   - Check one or more caches in the cache chart.
   - To add a column to the cache chart, check the column in the **Columns** list.
   - To remove a column from the cache chart, uncheck the column in the **Columns** list.
   - To change the order of columns, click on the column name in the cache chart and drag left or right.
   - To change the sort order of a column, click the triangle to the right of the column name in the cache chart.

Table 17-3 Columns and Properties for the Cache Chart Tab

Property	Description
`Cache Name`	The name of the cache.
`# of Objects`	The total number of objects in the selected cache.

Property	Description
`Cache Hits`	The total number times an Oracle Coherence cache client has accessed the selected cache.
`Cache Misses`	The total number times an Oracle Coherence cache client has accessed the selected cache and failed to find the desired object.
`Cache Prunes`	The total number times Oracle Coherence has reached its maximum size and had to delete some objects to return to a configured smaller size for the selected cache.
`Store Writes`	The total number of times Oracle Coherence has written to the selected cache's backing store.
`Store Reads`	The total number of times Oracle Coherence has read from the selected cache's backing store.
`Node Id`	The node identifier for the selected cache.
`Store Failures`	The total number of times the selected cache's backing store has failed to service an Oracle Coherence Store Write or Store Read.
`Average Hit Millis`	The average time (in milliseconds) taken by Oracle Coherence to find an object in the selected cache.
`Average Get Millis`	The average time (in milliseconds) taken by Oracle Coherence to retrieve an object from the selected cache.
`Average Miss Millis`	The average time (in milliseconds) taken by Oracle Coherence to determine that an object does not exist in the selected cache.
`Average Put Millis`	The average time (in milliseconds) taken by Oracle Coherence to add a new object to the selected cache.
`Cache Hits Millis`	The total time (in milliseconds) taken by Oracle Coherence to find an object in the selected cache.
`Cache Misses Millis`	The total time (in milliseconds) taken by Oracle Coherence to determine that an object does not exist in the selected cache.
`Cache Prunes Millis`	The total time (in milliseconds) taken by Oracle Coherence to delete some objects from the selected cache to return to a configured smaller size after having reached its maximum size.
`Store Read Millis`	The total time (in milliseconds) taken by Oracle Coherence to read from the selected cache's backing store.
`Store Write Millis`	The total time (in milliseconds) taken by Oracle Coherence to write to the selected cache's backing store.
`Total Gets Millis`	The total time (in milliseconds) taken by Oracle Coherence to retrieve objects from the selected cache.
`Total Puts Millis`	The total time (in milliseconds) taken by Oracle Coherence to add new objects to the selected cache.

5. Use the Cache Chart tab to compare any combination of the properties that Table 17-3 lists for one or more selected caches:

    - Check one or more caches in the cache chart.

- Drag a property from the **Drag Property to Watch** area to either graph to watch that property or, select the property and click **Add to Left** or **Add to Right**.

  You may specify at most one property per graph.

- A line is drawn for each selected cache. The legend below the graph distinguishes graph data for each selected cache by color.

- To watch a different property, drag the new property and drop it onto the graph or, select a different property and click **Add to Left** or **Add to Right**.

- To change the scale of a graph, drag the slider next to the vertical axis.

17.7 Tune Oracle Coherence

Using Oracle Event Processing Visualizer, you can view and modify a variety of Oracle Coherence parameters to fine tune Oracle Coherence performance.

Note:

Not all Oracle Coherence parameters can be modified. For more information, refer to your Oracle Coherence documentation.

To determine the root cause of a problem, you can monitor Oracle Coherence parameters. For more information, see:

- Monitor Nodes in an Oracle Coherence Cluster
- Monitor an Oracle Coherence Cache.

Tune Oracle Coherence:-

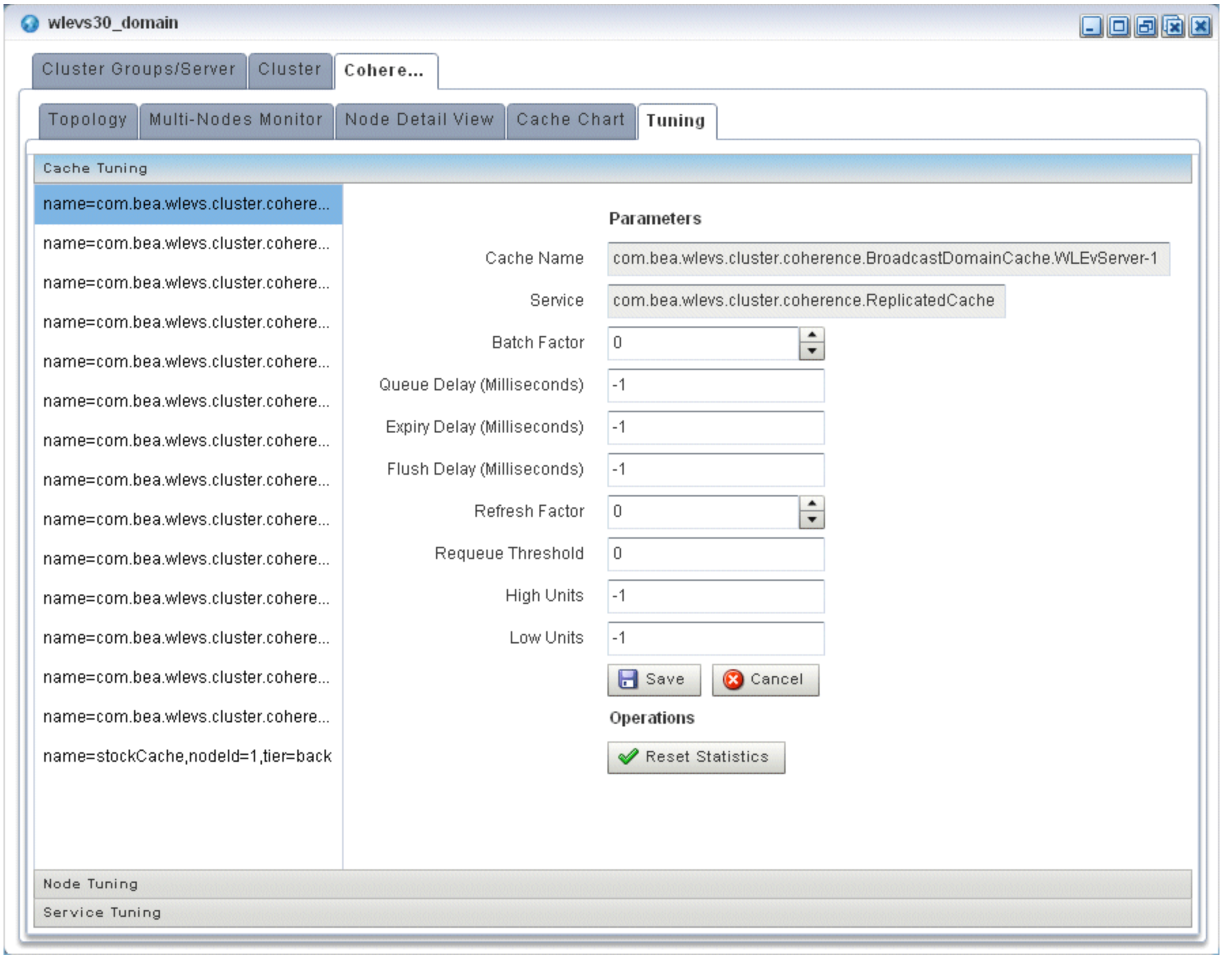
1. In the left pane, click the *Domain* node, where *Domain* refers to the name of your Oracle Event Processing domain.

2. In the right pane, click the **Coherence** tab.

3. In the right pane, click the **Tuning** tab.

   The Tuning tab appears as Figure 17-14 shows.

Figure 17-14 Tuning Tab: Cache Tuning



name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=com.bea.wlevs.cluster.cohere...
name=stockCache,nodeId=1,tier=back

Parameters

Cache Name	com.bea.wlevs.cluster.coherence.BroadcastDomainCache.WLEvServer-1
Service	com.bea.wlevs.cluster.coherence.ReplicatedCache
Batch Factor	0
Queue Delay (Milliseconds)	-1
Expiry Delay (Milliseconds)	-1
Flush Delay (Milliseconds)	-1
Refresh Factor	0
Requeue Threshold	0
High Units	-1
Low Units	-1

4. To tune parameters and invoke operations for caches, select the **Cache Tuning** accordion tab and select a cache as Figure 17-14 shows.

   Table 17-4 lists the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence `http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html`.

Table 17-4 Cache Tuning: Properties

Property	Description
`Cache Name`	The name of the selected cache (read-only).
`Service`	The service associated with this cache (read-only).

Property	Description
`Batch Factor`	The `BatchFactor` attribute is used to calculate the `soft-ripe` time for write-behind queue entries. A queue entry is considered to be `ripe` for a write operation if it has been in the write-behind queue for no less than the `QueueDelay` interval. The `soft-ripe` time is the point in time prior to the actual `ripe` time after which an entry will be included in a batched asynchronous write operation to the `CacheStore` (along with all other `ripe` and `soft-ripe` entries). This attribute is only applicable if asynchronous writes are enabled (that is, the value of the `QueueDelay` attribute is greater than zero) and the `CacheStore` implements the `storeAll()` method. The value of the element is expressed as a percentage of the `QueueDelay` interval. Valid values are doubles in the interval [0.0, 1.0]..
`Queue Delay (Milliseconds)`	The number of seconds that an entry added to a write-behind queue will sit in the queue before being stored via a `CacheStore`. Applicable only for `WRITE-BEHIND` persistence type.
`Expiry Delay (Milliseconds)`	The number of milliseconds that the MBeanServer will keep a remote model snapshot before refreshing.
`Flush Delay (Milliseconds)`	The number of milliseconds between cache flushes. Value of zero indicates that the cache will never flush.
`Refresh Factor`	The `RefreshFactor` attribute is used to calculate the `soft-expiration` time for cache entries. Soft-expiration is the point in time prior to the actual expiration after which any access request for an entry will schedule an asynchronous load request for the entry. This attribute is only applicable for a `ReadWriteBackingMap` which has an internal `LocalCache` with scheduled automatic expiration. The value of this element is expressed as a percentage of the internal `LocalCache` expiration interval. Valid values are doubles in the interval[0.0, 1.0]. If zero, refresh-ahead scheduling will be disabled.
`Requeue Threshold`	The maximum size of the write-behind queue for which failed `CacheStore` write operations are requeued. If zero, the write-behind requeueing will be disabled. Applicable only for `WRITE-BEHIND` persistence type.
`High Units`	The limit of the cache size measured in units. The cache will prune itself automatically once it reaches its maximum unit level. This is often referred to as the `high water mark` of the cache.
`Low Units`	The number of units to which the cache will shrink when it prunes. This is often referred to as a `low water mark` of the cache.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

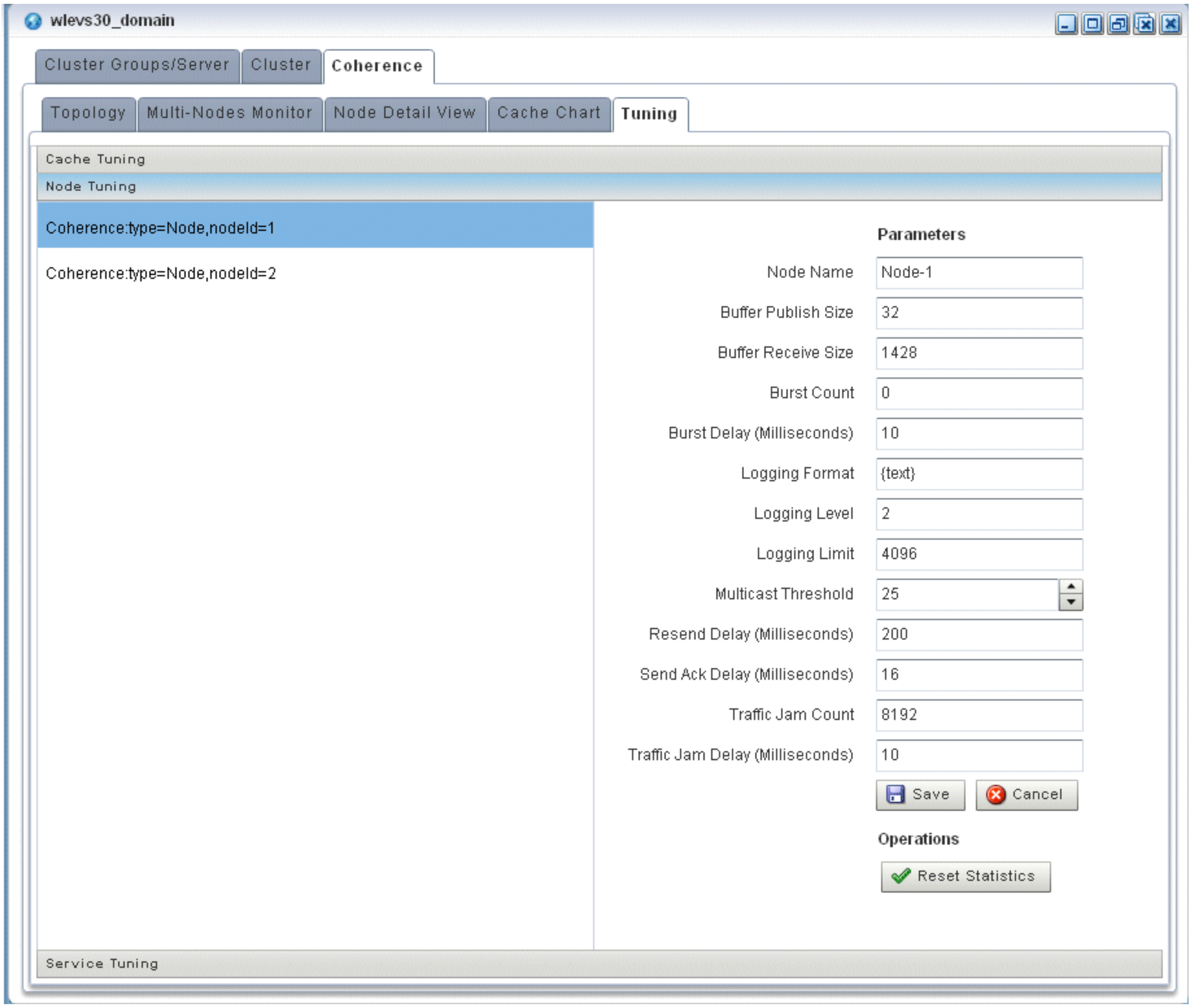
Table 17-5 lists the operations you can invoke for the selected cache.

Table 17-5 Cache Tuning: Operations

Operation	Description
`Reset Statistics`	Reset the cache statistics.

5. To tune parameters and invoke operations for cluster nodes, select the **Node Tuning** accordion tab and select a node as Figure 17-15 shows.

Figure 17-15 Tuning Tab: Node Tuning



Parameters

Node Name	Node-1
Buffer Publish Size	32
Buffer Receive Size	1428
Burst Count	0
Burst Delay (Milliseconds)	10
Logging Format	{text}
Logging Level	2
Logging Limit	4096
Multicast Threshold	25
Resend Delay (Milliseconds)	200
Send Ack Delay (Milliseconds)	16
Traffic Jam Count	8192
Traffic Jam Delay (Milliseconds)	10

Table 17-6 lists the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence `http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html`.

Table 17-6 Node Tuning: Properties

Property	Description
`Node Name`	The name of the selected node (read-only).
`Buffer Publish Size`	The buffer size of the unicast datagram socket used by the Publisher, measured in the number of packets. Changing this value at runtime is an inherently unsafe operation that will pause all network communications and may result in the termination of all cluster services.

Property	Description
`Buffer Receive Size`	The buffer size of the unicast datagram socket used by the Receiver, measured in the number of packets. Changing this value at runtime is an inherently unsafe operation that will pause all network communications and may result in the termination of all cluster services.
`Burst Count`	The maximum number of packets to send without pausing. Anything less than one (for example, zero) means no limit.
`Burst Delay (Milliseconds)`	The number of milliseconds to pause between bursts. Anything less than one (for example, zero) is treated as one millisecond.
`Logging Level`	Specifies which logged messages will be output to the log destination. Valid values are: • 0 only output without a logging severity level specified will be logged • 1 all the above plus errors • 2 all the above plus warnings • 3 all the above plus informational messages • 4..9 all the above plus internal debugging messages (the higher the number, the more the messages) • -1 no messages
`Logging Format`	Specifies how messages will be formatted before being passed to the log destination. The value of the message-format element is static text with the following replaceable parameters: • `{date}` the date/time format (to a millisecond) at which the message was logged • `{version}` the Oracle Coherence exact version and build details • `{level}` the logging severity level of the message • `{thread}` the thread name that logged the message • `{member}` the cluster member id (if the cluster is currently running) • `{text}` the text of the message
`Logging Limit`	The maximum number of characters that the logger daemon will process from the message queue before discarding all remaining messages in the queue. Valid values are integers in the range [0...]. Zero implies no limit.
`Multicast Threshold`	The percentage (0 to 100) of the servers in the cluster that a packet will be sent to, above which the packet will be multicasted and below which it will be unicasted.
`Resend Delay (Milliseconds)`	The minimum number of milliseconds that a packet will remain queued in the Publisher`s re-send queue before it is resent to the recipient(s) if the packet has not been acknowledged. Setting this value too low can overflow the network with unnecessary repetitions. Setting the value too high can increase the overall latency by delaying the re-sends of dropped packets. Additionally, change of this value may need to be accompanied by a change in `SendAckDelay` value.

Property	Description
`Send Ack Delay (Milliseconds)`	The minimum number of milliseconds between the queueing of an Ack packet and the sending of the same. This value should be not more then a half of the `ResendDelay` value.
`Traffic Jam Count`	The maximum total number of packets in the send and resend queues that forces the publisher to pause client threads. Zero means no limit.
`Traffic Jam Delay (Milliseconds)`	The number of milliseconds to pause client threads when a traffic jam condition has been reached. Anything less than one (for example, zero) is treated as one millisecond.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

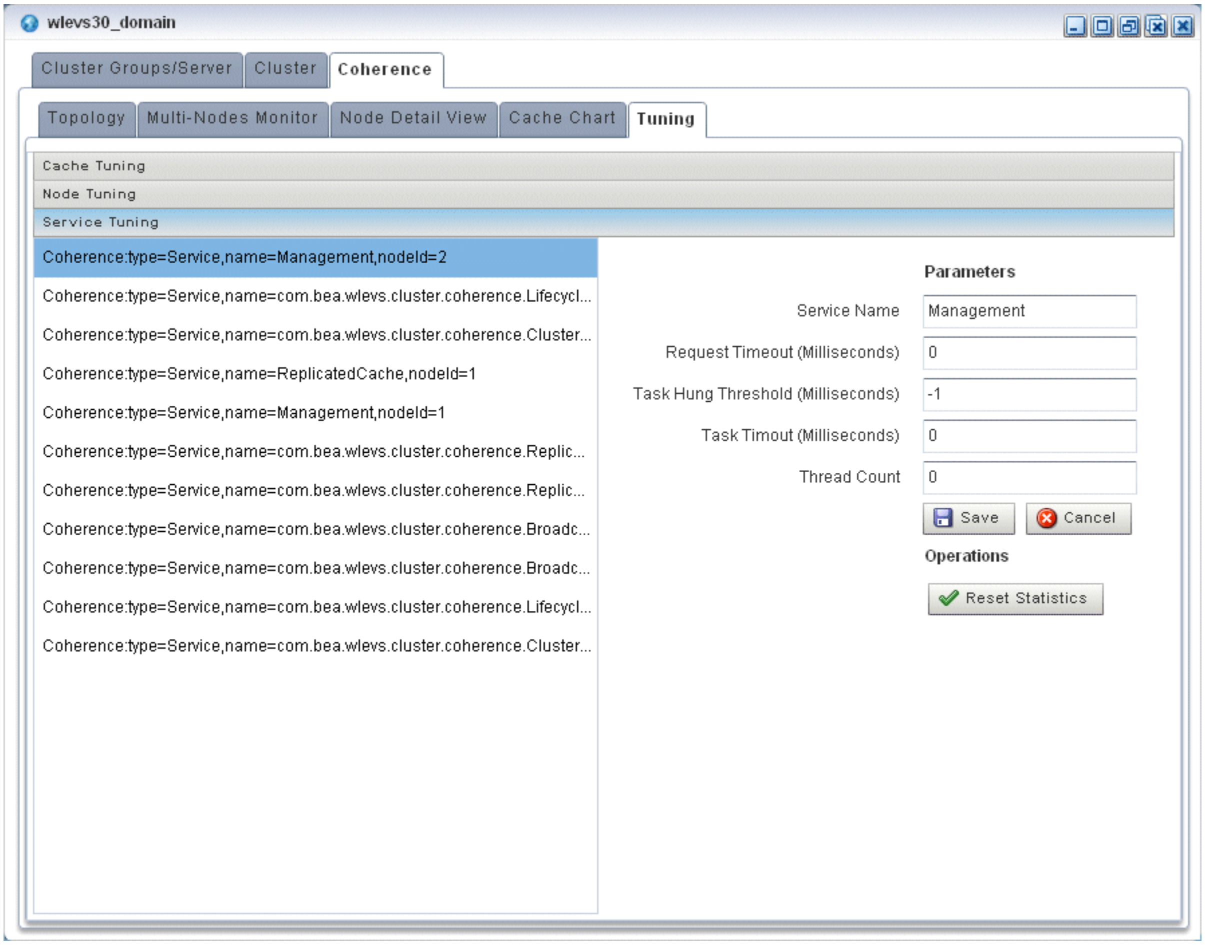
Table 17-5 lists the operations you can invoke for the selected node.

Table 17-7 Node Tuning: Operations

Operation	Description
`Reset Statistics`	Reset the cluster node statistics.

6. To tune parameters and invoke operations for nodes, select the **Node Tuning** accordion tab and select a node as Figure 17-16 shows.

Figure 17-16 Tuning Tab: Service Tuning



Coherence:type=Service,name=Management,nodeId=2
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Lifecycl...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Cluster...
Coherence:type=Service,name=ReplicatedCache,nodeId=1
Coherence:type=Service,name=Management,nodeId=1
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Replic...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Replic...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Broadc...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Broadc...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Lifecycl...
Coherence:type=Service,name=com.bea.wlevs.cluster.coherence.Cluster...

Service Name: Management
Request Timeout (Milliseconds): 0
Task Hung Threshold (Milliseconds): -1
Task Timout (Milliseconds): 0
Thread Count: 0

Table 17-8 the properties you can view and tune for the selected cache. All properties are based on Oracle Coherence `http://download.oracle.com/otn_hosted_doc/coherence/340/com/tangosol/net/management/Registry.html`.

Table 17-8 Service Tuning: Properties

Property	Description
`Service Name`	The name of the selected service (read-only).
`Request Timeout (Millis)`	The default timeout value in milliseconds for requests that can be timed-out (for example, implement the `com.tangosol.net.PriorityTask` interface), but do not explicitly specify the request timeout value.
`Task Hung Threshold (Millis)`	The amount of time in milliseconds that a task can execute before it is considered hung. Note that a posted task that has not yet started is never considered as hung.
`Task Timeout (Millis)`	The default timeout value in milliseconds for tasks that can be timed-out (for example, implement the `com.tangosol.net.PriorityTask` interface), but do not explicitly specify the task execution timeout value.
`Thread Count`	The number of threads in the service thread pool.

To apply changes, click the **Save** button.

To discard changes, click the **Cancel** button.

Table 17-9 lists the operations you can invoke for the selected service.

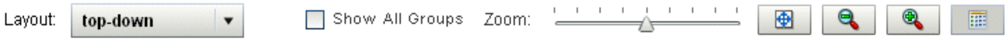
Table 17-9 Service Tuning: Operations

Operation	Description
`resetStatistics`	Reset the service statistics.

17.8 Manage the Cluster Topology Diagram

This section describes the various tools along the top of the Topology canvas as Figure 17-17 shows.

Figure 17-17 Cluster Topology Tools



Layout: top-down

You use these tools to manage the topology diagram of your cluster topology.

17.8.1 Choose Layout

Use the **Choose Layout** pull-down menu to select alternate ways of distributing cluster groups. You can choose any of:

- top-down
- organic
- hierarchic
- orthogonal

17.8.2 Show All Groups

Check the **Show All Groups** option to show all the cluster topology groups listed on the Cluster Groups/Server tab including the AllDomainMembers group and the per-server groups.

This option is unchecked by default to reduce clutter on the Cluster Topology tab and to allow you to focus on the most important groups.

17.8.3 Zoom

Use the Zoom slider to increase or decrease the zoom level. This is an alternative to using the Zoom In and Zoom Out buttons (see Zoom In and Zoom Out).

17.8.4 Fit Content

Click the **Fit Content** button to adjust the zoom level automatically to make all of the diagram visible in the current browser window.

17.8.5 Zoom In and Zoom Out

Click the **Zoom In** and **Zoom Out** buttons to change the zoom level. This is an alternative to using the Zoom slider (see Zoom).

17.8.6 Toggle Properties

Click the **Toggle Properties** button to alternately hide and show the properties table on the right.

18

Server Logs

This chapter describes how you can use Oracle Event Processing Visualizer to view and configure the Oracle Event Processing server logging service and component loggers.

This chapter includes the following sections:

- Configure the Server Logging Service
- Configure a Component Logger
- Query the Logs
- View Console Output

For more information, see Log Management.

18.1 Configure the Server Logging Service

Using the Oracle Event Processing Visualizer, you can configure the logging system of a selected Oracle Event Processing server. By default, this logging configuration applies to all components.

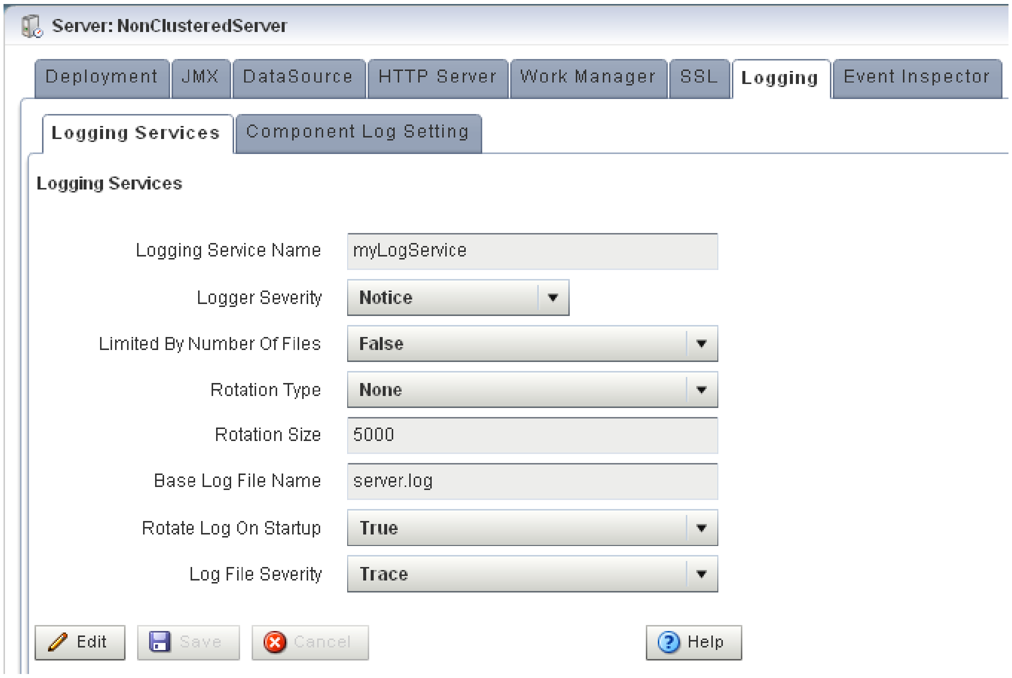
To configure component-specific logging options, see Configure a Component Logger.

Configure the Server Logging Service

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Logging Services** tab.

   The Logging Services tab appears as Figure 18-1 shows.

Figure 18-1 Logging Services Tab



Logging Services

Logging Service Name	myLogService
Logger Severity	Notice
Limited By Number Of Files	False
Rotation Type	None
Rotation Size	5000
Base Log File Name	server.log
Rotate Log On Startup	True
Log File Severity	Trace

4. Click **Edit**.

5. Edit the dialog as Table 18-1 describes.

Table 18-1 Logging Services Options

Attribute	Description
`Logging Service Name`	Specify the name of the logging service as the Oracle Event Processing server `config.xml` file specifies.
`Logger Severity`	Specify the default log severity at which Oracle Event Processing server logs messages.
`Limited By Number Of Files`	Specify whether or not to keep old log files indefinitely: • `true`: keep only the last 7 log files. • `false`: keep all log files.
`Rotation Type`	Specify when Oracle Event Processing server should close the existing log file and open a new one: • `Size`: Oracle Event Processing server closes the existing log file when it reaches the size (in KB) you enter in the `Rotation Size` field. • `Time`: Oracle Event Processing server closes the existing log file at the time you enter in the numeric entry field below the Rotation Type pull-down menu in `k:mm` format, where k is the hour specified in 24 hour notation and `mm` is the minutes. Default-Value: 00:00. • `None`: Oracle Event Processing server never closes the existing log file and allows it to grow to an unlimited size.
`Rotation Size`	When you specify `Rotation Type` as `Size`, enter the maximum log file size in KB.

Attribute	Description
`Base Log File Name`	The name of the base log file. Default: `server.log`.
`Rotate Log On Startup`	Specify whether or not Oracle Event Processing server closes the existing log file and creates a new one each time the Oracle Event Processing server starts up: • `true`: Oracle Event Processing server closes the existing log file and creates a new one each time the Oracle Event Processing server starts up. • `false`: Oracle Event Processing server does not close the existing log file each time the Oracle Event Processing server starts up.
Log File Severity	Specify the default log file severity. Oracle Event Processing server (and, by default, all components), log messages at this severity: • `Emergency` • `Alert` • `Critical` • `Error` • `Warning` • `Notice` • `Info` • `Debug` • `Trace`

6. Click the **Save** button to commit your changes.

18.2 Configure a Component Logger

Using the Oracle Event Processing Visualizer, you can configure the logging properties of a selected component. Table 18-2 lists the components that are configured to log at the `Notice` log level, by default.

This section describes how:

- Add a Component Logger
- Change the Logging Level of a Component
- To delete a component logger:

To configure logging properties that apply to all components, see Configure the Server Logging Service.

Add a Component Logger

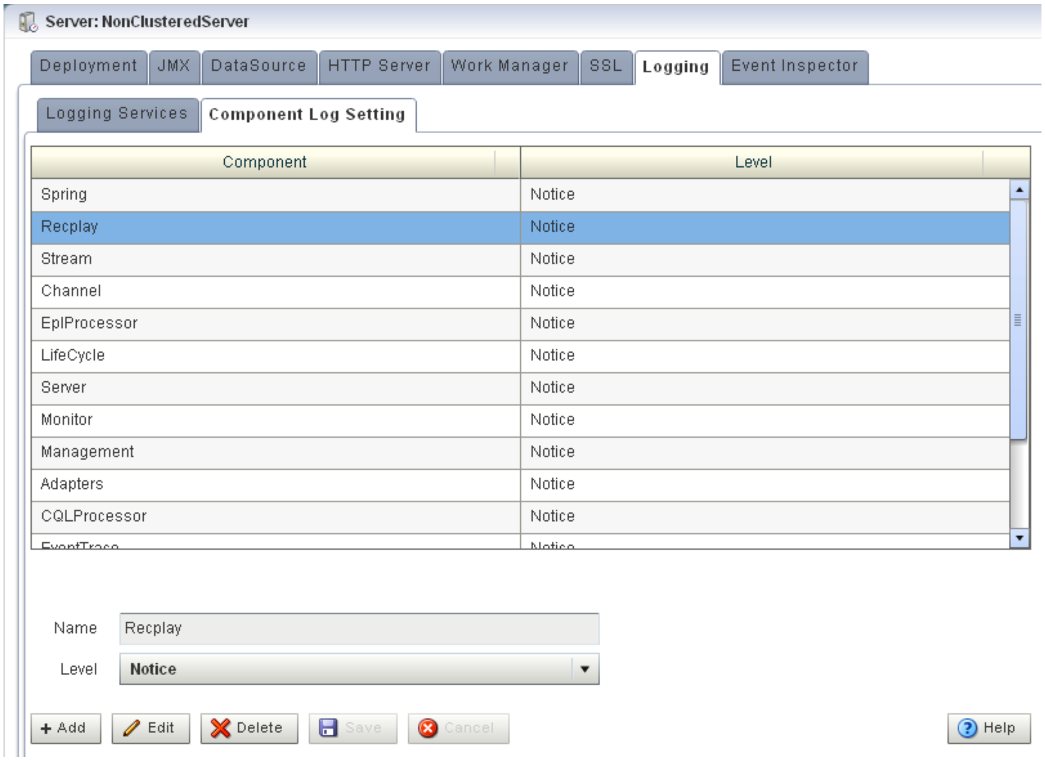
1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. In the right pane, click the **Logging** tab.

3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 18-2 show.

Figure 18-2 Component Log Setting Tab



Component	Level
Spring	Notice
Recplay	Notice
Stream	Notice
Channel	Notice
EplProcessor	Notice
LifeCycle	Notice
Server	Notice
Monitor	Notice
Management	Notice
Adapters	Notice
CQLProcessor	Notice
EventTrace	Notice

Name: Recplay

Level: Notice

4. Click **Add**.

5. Enter the name of the component in the **Name** field.

   You can choose any of the following:

   - **Component name**: a component name constant exactly as Table 18-2 lists.

Table 18-2 Logging Component Name Constants

Component Name	Description
`Adapters`	Applies to log messages from adapter instances running on the Oracle Event Processing server.
`Cache`	Applies to log messages from caching systems and cache instances running on the Oracle Event Processing server.
`Channel`	Applies to log messages from channels running on the Oracle Event Processing server.
`CQLProcessor`	Applies to log messages from Oracle CQL processors running on the Oracle Event Processing server.
`CQLServer`	Applies to log messages from the CQL Engine, which is at the core of each CQLProcessor.
`CQLServerTrace`	Applies to log messages from the CQL Engine, which is at the core of each CQLProcessor

Component Name	Description
`Ede`	Applies to log messages from the Event-Driven Environment, the Oracle Event Processing server event-dispatching infrastructure.

`EventTrace`	When set to `Info` or `Debug`, allows you to trace events as they flow through the EPN for all applications. You can dynamically change the severity of this log key using Oracle Event Processing Visualizer.
	At the `Info` severity, you see log messages like this (although likely displayed on one line):
	`<May 26, 2009 5:53:49 PM PDT> <Info> <EventTrace>`
	`<BEA-000000>`
	`    <Application [helloworld], Stage`
	`[helloworldOutputChannel]`
	`    received insert event>`
	At the `Debug` severity, the log messages include details of the event:
	`<May 26, 2009 6:02:34 PM PDT> <Debug> <EventTrace>`
	`<BEA-000000>`
	`    <Application [helloworld], Stage`
	`[helloworldOutputChannel]`
	`    received insert event [HelloWorldEvent: HelloWorld -`
	`the`
	`    current time is: 6:02:34 PM]>`

`Lifecycle`	Applies to log messages from Oracle Event Processing server and application life cycle operations.
`Management`	Applies to log messages from Oracle Event Processing server general JMX-related management API operations.
`Monitor`	Applies to log messages from the Oracle Event Processing server monitoring service.
`Recplay`	Applies to log messages from Oracle Event Processing server event recording and playback operations.
`Spring`	Applies to log messages from Spring container operations.
`Stream`	Applies to log messages from stream instances running on the Oracle Event Processing server.

- **Application name**: the module name of any Oracle Event Processing server or user-defined application. For example: `sample.HelloWorld`.
- **Package name**: the name of any Oracle Event Processing server or user-supplied Java package. For example: `com.bea.wlevs.ede`.
- **Class name**: the fully qualified name of any Oracle Event Processing server or user-defined class. For example: `com.bea.wlevs.cep.core.EPRuntimeImpl`.

6. Select the severity level from the **Level** drop-down menu:

- `Emergency`
- `Alert`
- `Critical`
- `Error`
- `Warning`
- `Notice`
- `Info`
- `Debug`
- `Trace`

7. Click **Save**.

   The component and the severity level is displayed in the table.

Change the Logging Level of a Component

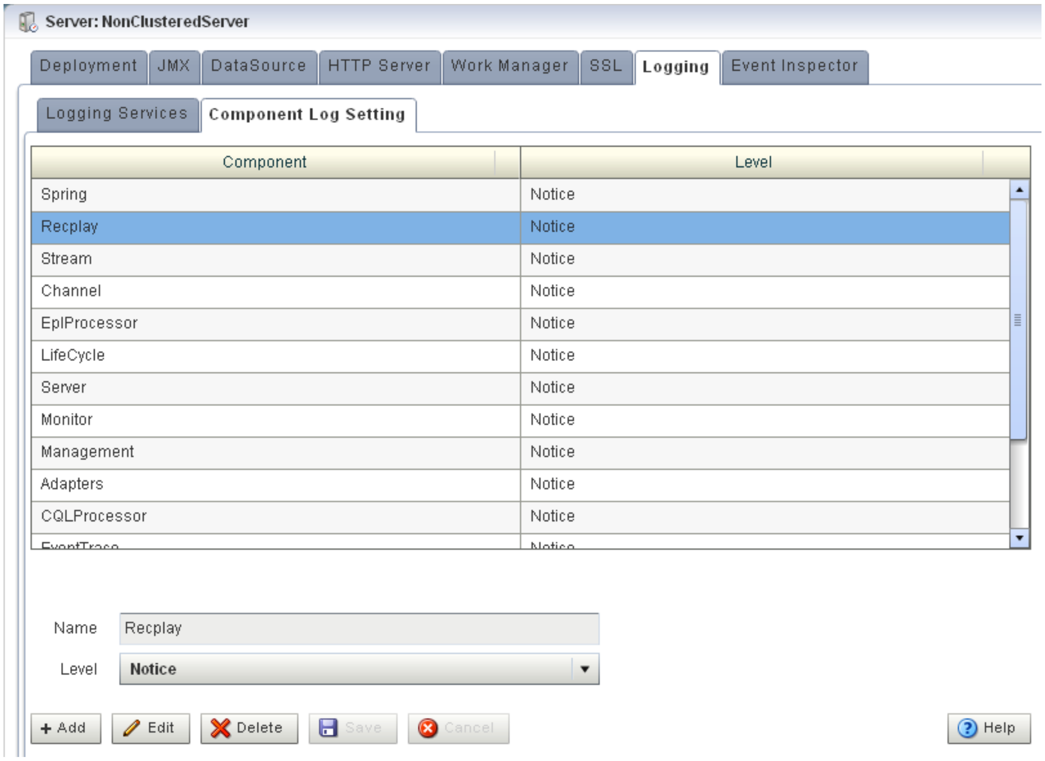
1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. In the right pane, click the **Logging** tab.

3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 18-3 show.

Figure 18-3 Component Log Setting Tab



Component	Level
Spring	Notice
Recplay	Notice
Stream	Notice
Channel	Notice
EplProcessor	Notice
LifeCycle	Notice
Server	Notice
Monitor	Notice
Management	Notice
Adapters	Notice
CQLProcessor	Notice
EventTrace	Notice

Name: Recplay

Level: Notice

4. Select the component entry in the table.
5. Click **Edit**.
6. Select the severity level from the **Level** drop-down menu:
   - `Emergency`
   - `Alert`
   - `Critical`
   - `Error`
   - `Warning`
   - `Notice`
   - `Info`
   - `Debug`
   - `Trace`
7. Click **Save**.

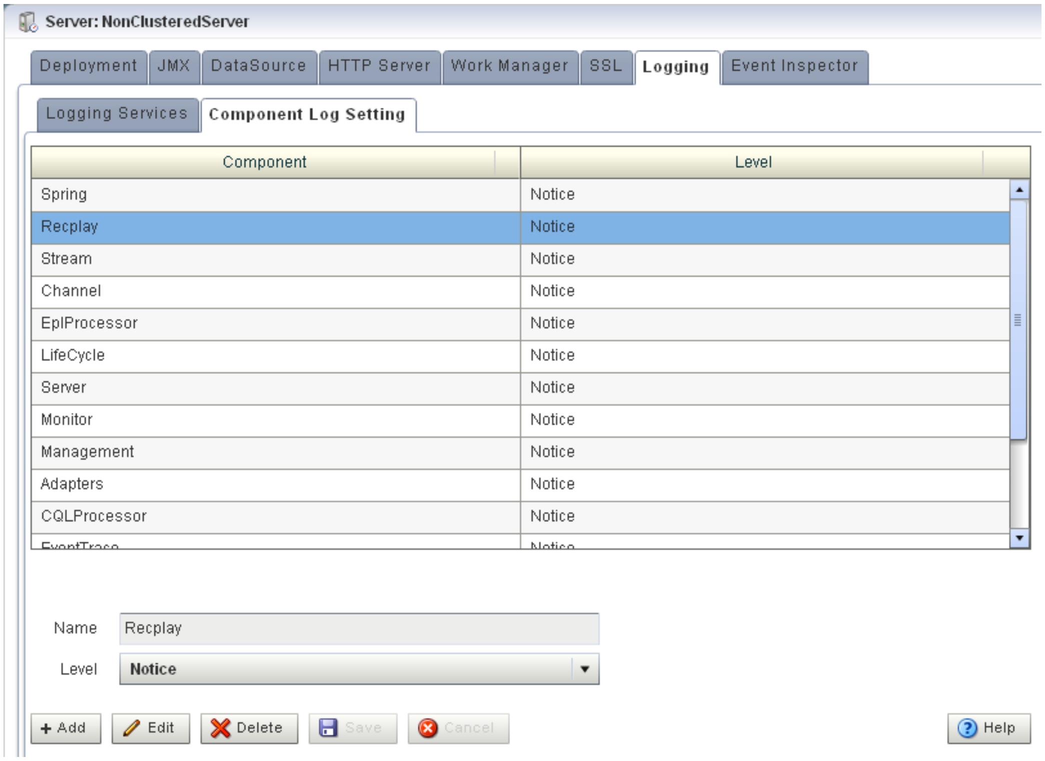
   The new severity level is displayed in the table.

To delete a component logger:

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.
2. In the right pane, click the **Logging** tab.
3. Click the **Component Log Setting** tab.

   The Component Log Setting tab appears as Figure 18-4 show.

Figure 18-4 Component Log Setting Tab



Component	Level
Spring	Notice
Recplay	Notice
Stream	Notice
Channel	Notice
EplProcessor	Notice
LifeCycle	Notice
Server	Notice
Monitor	Notice
Management	Notice
Adapters	Notice
CQLProcessor	Notice
EventTrace	Notice

Name: Recplay

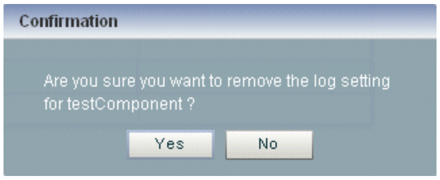
Level: Notice

4. Select the component entry in the table.

5. Click **Delete**.

   A confirmation dialog appears as Figure 18-5 shows.

Figure 18-5 Component Log Configuration Delete Confirmation



Are you sure you want to remove the log setting for testComponent ?

6. Click **Yes**.

   The component logger is removed from the table.

18.3 Query the Logs

Using the Oracle Event Processing Visualizer, you can run queries on the logs of a selected Oracle Event Processing server. The Query Logs feature allows you to view the information in the selected log files.

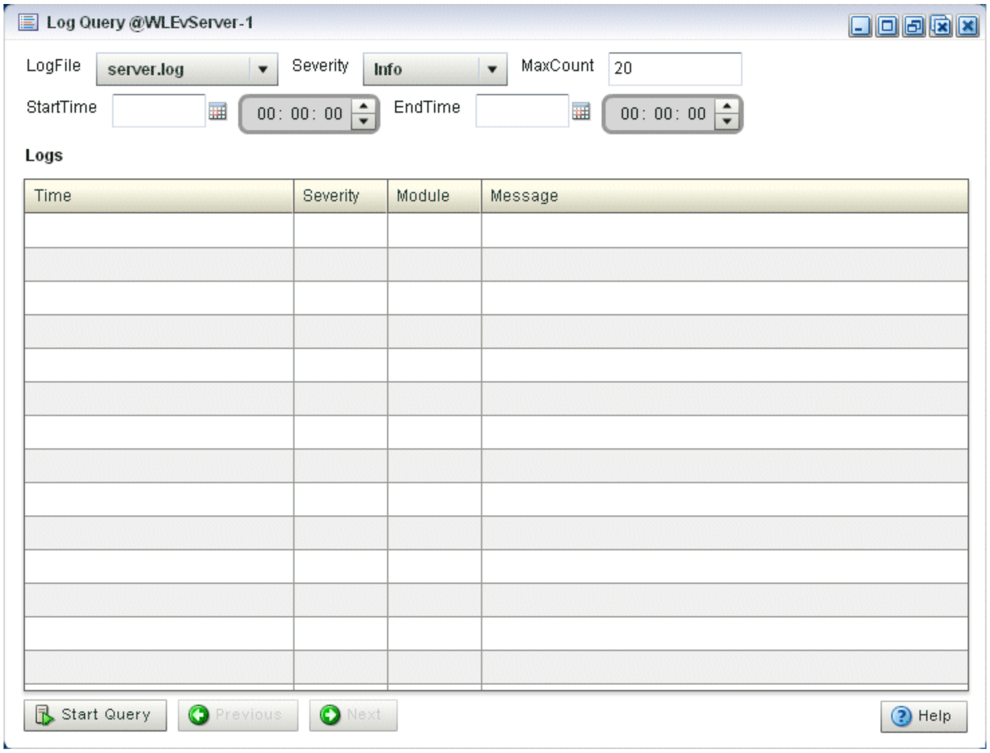
Query the Logs

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. Click **Services** and then click **Log Query**.

   The Log Query screen is displayed, as shown in Figure 18-6.

   Figure 18-6 Log Query Screen

   Log Query @WLEvServer-1

   LogFile: server.log

   Severity: Info

   MaxCount: 20

   StartTime: 00: 00: 00

   EndTime: 00: 00: 00

   Logs

Time	Severity	Module	Message

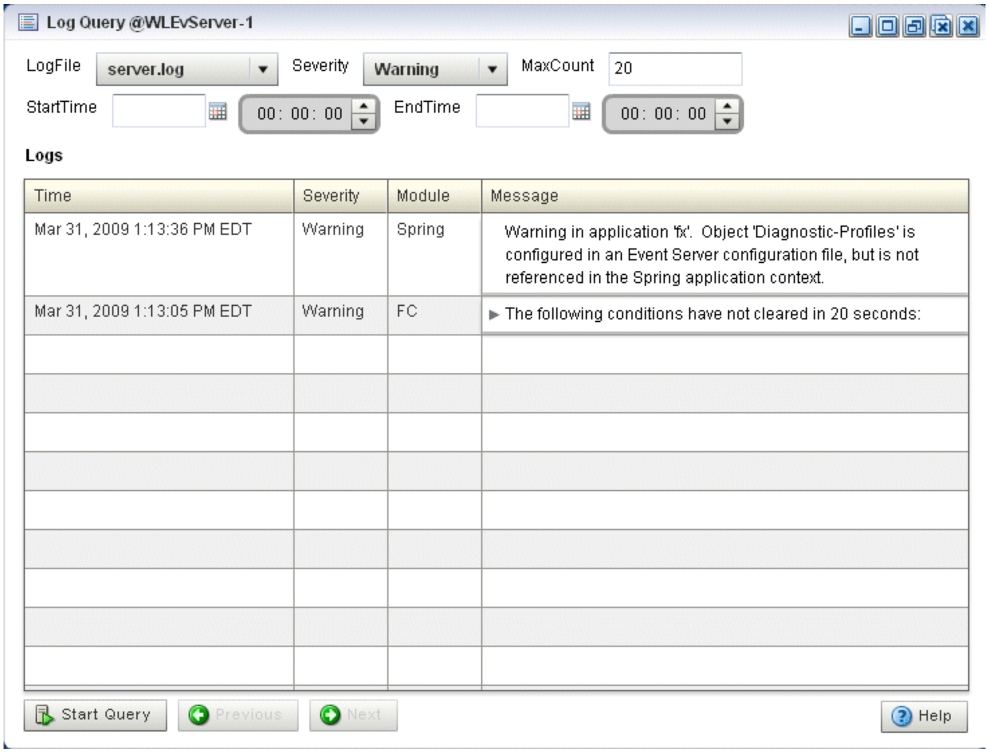
3. Select the following options in the Log Query screen:

   - `LogFile`: Select the log file you want to query from the drop-down.
   - `Severity`: Select the severity of the log file that you selected in the LogFile drop-down.
   - `MaxCount`: Enter the maximum count of rows of log you wish to view for the selected log file.
   - `StartTime`: Select the time when the query should start running.
   - `EndTime`: Select the time when the query should stop.

4. Click **Start Query**.

   All the log messages that match your query criteria for the selected log file of the selected Oracle Event Processing Server are displayed in the Logs pane, as shown in Figure 18-7.

Figure 18-7 Log Query Screen - Logs Displayed



LogFile: server.log
Severity: Warning
MaxCount: 20
StartTime: 00: 00: 00
EndTime: 00: 00: 00

Logs

Time	Severity	Module	Message
Mar 31, 2009 1:13:36 PM EDT	Warning	Spring	Warning in application 'fx'. Object 'Diagnostic-Profiles' is configured in an Event Server configuration file, but is not referenced in the Spring application context.
Mar 31, 2009 1:13:05 PM EDT	Warning	FC	▶ The following conditions have not cleared in 20 seconds:

5. If more messages are returned than can be displayed in the Logs pane, the Previous and Next buttons activate. Click the **Previous** and **Next** buttons to page through the messages.

18.4 View Console Output

Using the Oracle Event Processing Visualizer, you can view the console output of a selected Oracle Event Processing server.

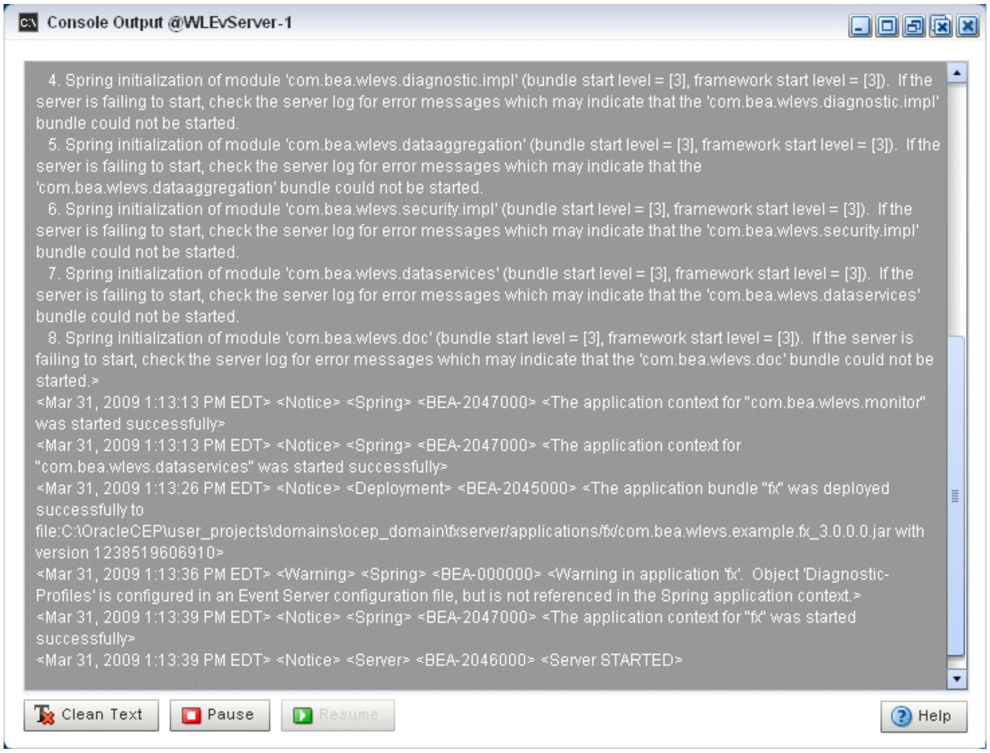
View Console Output

1. In the left pane, click the *Domain* > *Server* node, where *Domain* refers to the name of your Oracle Event Processing domain and *Server* refers to the name of the server instance.

2. Click **Services** and then click **Console Output**.

   The Console Output screen is displayed, as shown in Figure 18-8.

Figure 18-8 Sample Console Output for HelloWorld Application



```
  4. Spring initialization of module 'com.bea.wlevs.diagnostic.impl' (bundle start level = [3], framework start level = [3]).  If the
server is failing to start, check the server log for error messages which may indicate that the 'com.bea.wlevs.diagnostic.impl'
bundle could not be started.
  5. Spring initialization of module 'com.bea.wlevs.dataaggregation' (bundle start level = [3], framework start level = [3]).  If the
server is failing to start, check the server log for error messages which may indicate that the
'com.bea.wlevs.dataaggregation' bundle could not be started.
  6. Spring initialization of module 'com.bea.wlevs.security.impl' (bundle start level = [3], framework start level = [3]).  If the
server is failing to start, check the server log for error messages which may indicate that the 'com.bea.wlevs.security.impl'
bundle could not be started.
  7. Spring initialization of module 'com.bea.wlevs.dataservices' (bundle start level = [3], framework start level = [3]).  If the
server is failing to start, check the server log for error messages which may indicate that the 'com.bea.wlevs.dataservices'
bundle could not be started.
  8. Spring initialization of module 'com.bea.wlevs.doc' (bundle start level = [3], framework start level = [3]).  If the server is
failing to start, check the server log for error messages which may indicate that the 'com.bea.wlevs.doc' bundle could not be
started.>
<Mar 31, 2009 1:13:13 PM EDT> <Notice> <Spring> <BEA-2047000> <The application context for "com.bea.wlevs.monitor"
was started successfully>
<Mar 31, 2009 1:13:13 PM EDT> <Notice> <Spring> <BEA-2047000> <The application context for
"com.bea.wlevs.dataservices" was started successfully>
<Mar 31, 2009 1:13:26 PM EDT> <Notice> <Deployment> <BEA-2045000> <The application bundle "fx" was deployed
successfully to
file:C:\OracleCEP\user_projects\domains\ocep_domain\fxserver/applications/fx/com.bea.wlevs.example.fx_3.0.0.0.jar with
version 1238519606910>
<Mar 31, 2009 1:13:36 PM EDT> <Warning> <Spring> <BEA-000000> <Warning in application 'fx'.  Object 'Diagnostic-
Profiles' is configured in an Event Server configuration file, but is not referenced in the Spring application context.>
<Mar 31, 2009 1:13:39 PM EDT> <Notice> <Spring> <BEA-2047000> <The application context for "fx" was started
successfully>
<Mar 31, 2009 1:13:39 PM EDT> <Notice> <Server> <BEA-2046000> <Server STARTED>
```

3. Click the **Clean Text** button to clear the text in the Console Output window.
4. Click the **Pause** button to pause the output of console messages.
5. Click the **Resume** button to resume the output of console messages.

Part IV

Security Tasks

Security Tasks contains the following chapters:

- Security Tasks
- User Accounts and Passwords
- User Groups
- Task Roles
- HTTP Publish-Subscribe Server Security
- SSL Configuration.

19

Security Tasks

This chapter introduces security tasks you can perform with Oracle Event Processing Visualizer, including managing users, groups, and roles, as well as managing HTTP publish-subscribe server channel security and SSL.

This chapter includes the following sections:

- User, Group, and Role Management
- HTTP Publish-Subscribe Server Channel Security Management
- SSL Management.

See also *Administering Oracle Event Processing*.

19.1 User, Group, and Role Management

Oracle Event Processing uses role-based authorization control to secure the Oracle Event Processing Visualizer and the `wlevs.Admin` command-line utility. There are a variety of default out-of-the-box security groups. You can add users to different groups to give them the different roles.

Administrators who use Oracle Event Processing Visualizer, `wlevs.Admin,` or any custom administration application that uses JMX to connect to an Oracle Event Processing instance use role-based authorization to gain access.

You can also use role-based authorization to control access to the HTTP publish-subscribe server.

There are two types of role:

- Application roles: application roles grant users the permission to access various Oracle CQL applications deployed to the Oracle Event Processing server. You can create application roles and associate them with the task roles that Oracle Event Processing provides.

  By default, administrator users can access any application and non-administration users cannot access any applications. Before a none-administration user can access an application, an administration user must grant the user the associated application role.

- Task roles: task roles grant users the permission to perform various tasks with the applications their application role authorizes them to access. Oracle Event Processing provides the default task roles that Table 19-1 describes.

Users that successfully authenticate themselves when using Oracle Event Processing Visualizer or `wlevs.Admin` are assigned roles based on their group membership, and then subsequent access to administrative functions is restricted according to the roles held by the user. Anonymous users (non-authenticated users) will not have any access to the Oracle Event Processing Visualizer or `wlevs.Admin`.

When an administrator uses the Configuration Wizard to create a new domain, they enter an administrator user that will be part of the wlevsAdministrators group. By default, this information is stored in a file-based provider filestore. The password is hashed using the SHA-256 algorithm. The default administrator user is named oepadmin with password welcome1.

Table 19-1 describes the default Oracle Event Processing task roles available right after the creation of a new domain, as well as the name of the groups that are assigned to these roles.

Table 19-1 Default Oracle Event Processing Task Roles and Groups

Task Role	Group	Privileges
Admin	wlevsAdministrators	Has all privileges of all the preceding roles, and permission to: • Create users and groups • Configure HTTP publish-subscribe security • Change the system configuration, such as Jetty, work manager, and so on.
ApplicationAdmin	wlevsApplicationAdmins	Has all Operator privileges except the permission to update the queries, which is allowed only with the Admin and BusinessUser roles. ApplicationAdmin includes permission to update the configuration of any deployed application.
BusinessUser	wlevsBusinessUsers	Has all Operator privileges and permission to update the Oracle CQL rules associated with the processor of a deployed application.
Deployer	wlevsDeployers	Has all Operator privileges and permission to deploy, undeploy, update, suspend, and resume any deployed application.
Monitor	wlevsMonitors	Has all Operator privileges and permission to monitor (diagnostic profile), record, playback, and to perform event traces and event injections.
Operator	wlevsOperators	Has read-only access to all server resources, services, and deployed applications.

Use Oracle Event Processing Visualizer to create a group for an existing domain and to assign roles to the group. When you assign a user to the group, the roles give that user the privileges to perform the actions allowed by the roles.

Using Oracle Event Processing Visualizer, you can:

- User Accounts and Passwords
- User Groups
- Task Roles.

19.2 HTTP Publish-Subscribe Server Channel Security Management

Oracle Event Processing provides an HTTP Publish-Subscribe Server (HTTP pub-sub server): a mechanism whereby Web clients subscribe to channels (similar to a topic in

JMS) and then publish messages to these channels using asynchronous messages over HTTP and subscribe to these channels to receive messages as they become available.

Using Oracle Event Processing Visualizer, you can specify which users can access HTTP publish-subscribe server channels.

For more information, see:

- HTTP Publish-Subscribe Server Security
- *Administering Oracle Event Processing*.

19.3 SSL Management

Oracle Event Processing provides one-way Secure Sockets Layer (SSL) to secure network traffic between Oracle Event Processing Visualizer and Oracle Event Processing server instances, between the Oracle Event Processing server instances of a multiserver domain, and between the `wlevs.Admin` command-line utility and Oracle Event Processing server instances.

You configure SSL in the Oracle Event Processing server `config.xml` file. By default, the Configuration Wizard creates the `config.xml` file in the *`Oracle/Middleware/my_oep/user_projects/domains/DOMAIN_DIR/servername/`*`config` directory.

For more information, see:

- SSL Configuration
- *Administering Oracle Event Processing*.

20

User Accounts and Passwords

This chapter describes how you can use Oracle Event Processing Visualizer to create, edit, and delete user accounts, including user passwords.

This chapter includes the following sections:

For more information, see User, Group, and Role Management.

20.1 Create a User

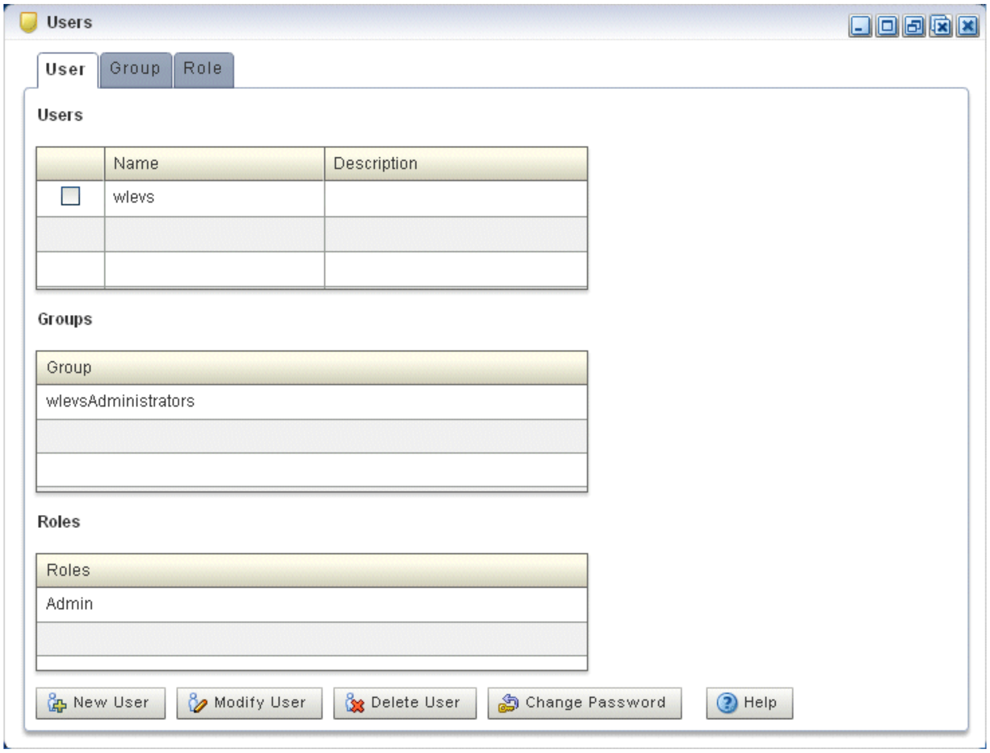
Using the Oracle Event Processing Visualizer, you can create a user.

Create a User

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **User** tab.

   The User tab appears as Figure 20-1 shows.

Figure 20-1 User Tab



Users

	Name	Description
□	wlevs	

Groups

Group
wlevsAdministrators

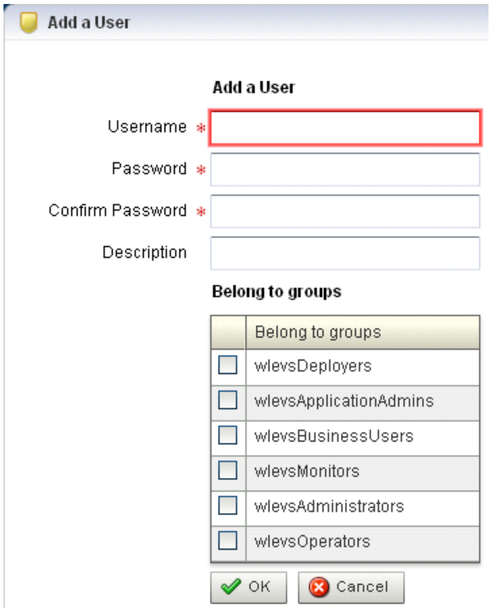
Roles

Roles
Admin

4. Click the **New User** button at the bottom of the right pane.

   The Add a User panel appears as Figure 20-2 shows.

Figure 20-2 Add a User Panel



Add a User

Username *	
Password *	
Confirm Password *	
Description	

Belong to groups

	Belong to groups
□	wlevsDeployers
□	wlevsApplicationAdmins
□	wlevsBusinessUsers
□	wlevsMonitors
□	wlevsAdministrators
□	wlevsOperators

5. Configure the Add a User panel as Table 20-1 describes.

Table 20-1 Add a User Panel Attributes

Attribute	Description
`Username`	Enter the name of the user.
`Password`	Enter the password for this user. Passwords must be at least 6 characters in length.
`Confirm Password`	Re-enter the password for this user.
`Description`	An optional description for this user.
`Belong to groups`	Check one or more groups to which the user belongs. The user inherits the privileges of the roles you assign to the groups. You must assign a user to at least one group. For more information, see User Groups.

6. Click **OK**.

   When the account has been successfully created, a confirmation message appears momentarily.

20.2 Edit a User

You can edit the configuration of existing users.

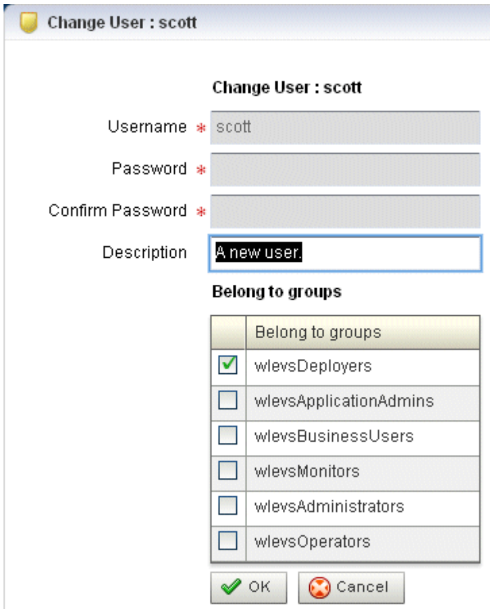
To edit user passwords, see Change the User Password.

Edit a User

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **User** tab.
4. In the Users table, check the box to the left of the name of the user that you want to modify.
5. Click the **Modify User** button at the bottom of the right pane.

   The Change User panel appears as Figure 20-3 shows.

Figure 20-3 Change User Panel



Change User : scott

Username * scott

Password *

Confirm Password *

Description A new user.

Belong to groups

□	Belong to groups
☑	wlevsDeployers
□	wlevsApplicationAdmins
□	wlevsBusinessUsers
□	wlevsMonitors
□	wlevsAdministrators
□	wlevsOperators

6. Configure the Change User panel as Table 20-2 describes.

Table 20-2 Change User Panel Attributes

Attribute	Description
`Username`	The name of the user. This is a read-only field.
`Password`	This field is blank and read-only. To modify the password, see Change the User Password.
`Confirm Password`	This field is blank and read-only. To modify the password, see Change the User Password.
`Description`	Modify the optional description for this user.
`Belong to groups`	Modify the groups to which the user belongs by checking or unchecking one or more of the groups displayed. Check one or more groups to which the user belongs. The user inherits the privileges of the roles you assign to the groups You must assign a user to at least one group. For more information, see User Groups.

7. Click **OK**.

20.3 Change the User Password

You can change the password for existing users.

To change other user configuration options, see Edit a User.

Change the User Password

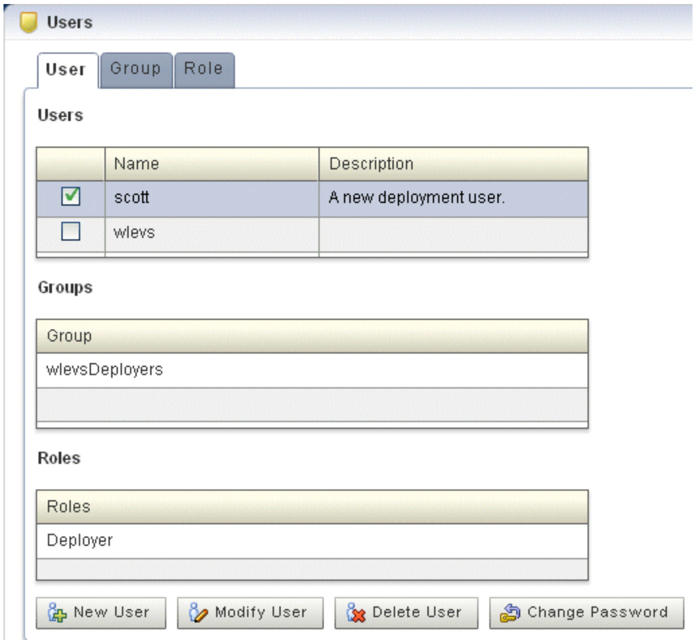
1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.

2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.

3. In the right pane, click the **User** tab.

4. In the Users table, check the boxes to the left of the name of the users that you want to delete as Figure 20-4 shows.

Figure 20-4 Selecting a User



Users

Users

	Name	Description
☑	scott	A new deployment user.
□	wlevs	

Groups

Group
wlevsDeployers

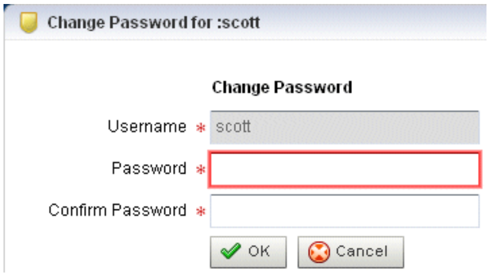
Roles

Roles
Deployer

5. Click the **Change Password** button at the bottom of the right pane.

   The Change Password panel appears as Figure 20-5 shows.

Figure 20-5 Change Password Panel



Change Password for :scott

Change Password

Username * scott

Password *

Confirm Password *

6. Configure the Change Password panel as Table 20-2 describes.

Table 20-3 Change User Panel Attributes

Attribute	Description
`Username`	The name of the user. This is a read-only field.
`Password`	This field is blank. To modify the password, see Change the User Password.
`Confirm Password`	This field is blank. To modify the password, see Change the User Password.

7. Click **OK**.

When the password has been successfully changed, a confirmation message appears momentarily.

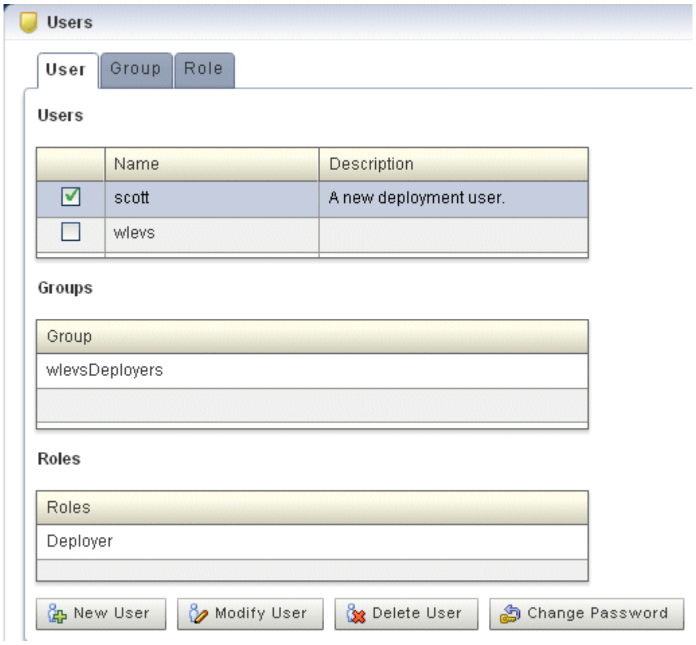
20.4 Delete a User

You can delete existing users. However, you cannot delete the default administrator user, which is the administrator user originally configured for the domain when it was created with the Configuration Wizard.

Delete a User

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **User** tab.
4. In the Users table, check the boxes to the left of the name of the users that you want to delete as Figure 20-6 shows.

Figure 20-6 Selecting a User



Users

	Name	Description
☑	scott	A new deployment user.
□	wlevs	

Groups

Group
wlevsDeployers

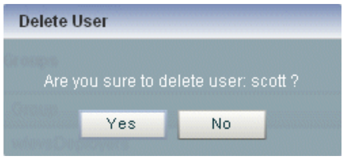
Roles

Roles
Deployer

5. Click the **Delete User** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 20-7 shows.

Figure 20-7 Delete User Dialog



Are you sure to delete user: scott ?

6. Click **Yes**.

When the account has been successfully deleted, a confirmation message appears momentarily.

21

User Groups

This chapter describes how you can use Oracle Event Processing Visualizer to create, edit, and delete user groups, including editing group membership and mapping groups to roles.

This chapter includes the following sections:

- Create a Group
- Change Group Roles
- Delete a Group

See also User, Group, and Role Management. To change the groups to which a user is assigned, see Edit a User.

21.1 Create a Group

Oracle Event Processing is configured by default with a set of groups that are in turn mapped to roles. See User, Group, and Role Management for details.

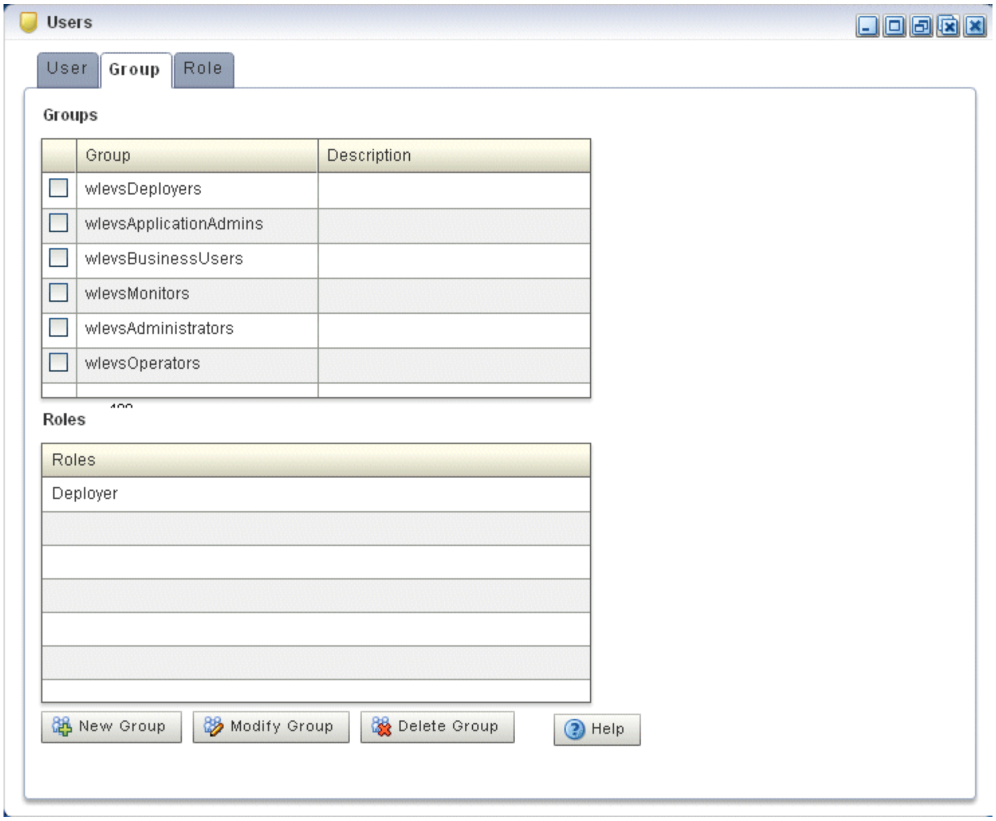
This section describes how to create a new group.

Create a Group

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **Group** tab.

   The Group tab appears as Figure 21-1 shows.

Figure 21-1 Group Tab



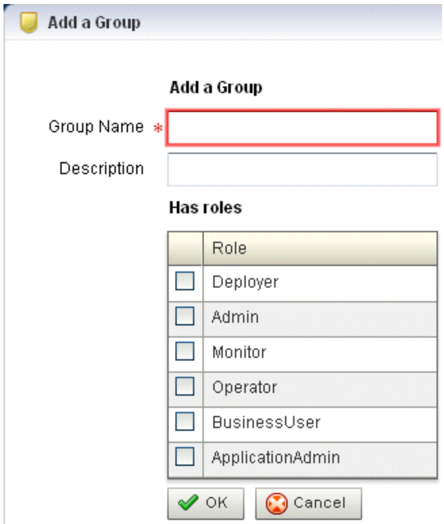
	Group	Description
□	wlevsDeployers	
□	wlevsApplicationAdmins	
□	wlevsBusinessUsers	
□	wlevsMonitors	
□	wlevsAdministrators	
□	wlevsOperators	

Roles
Deployer

4. Click the **New Group** button at the bottom of the right pane.

   The Add a Group panel appears as Figure 21-2 shows.

Figure 21-2 Add a Group Panel



	Role
□	Deployer
□	Admin
□	Monitor
□	Operator
□	BusinessUser
□	ApplicationAdmin

5. Configure the Add a Group panel as Table 21-1 describes.

Table 21-1 Add a Group Panel Attributes

Attribute	Description
`Group Name`	Enter the name of the group.
`Description`	An optional description for this user.
`Has roles`	Check one or more roles to which the group maps. You must select at least one role. For more information, see Task Roles.

6. Click **OK**.

   When the group has been successfully created, a confirmation message appears momentarily.

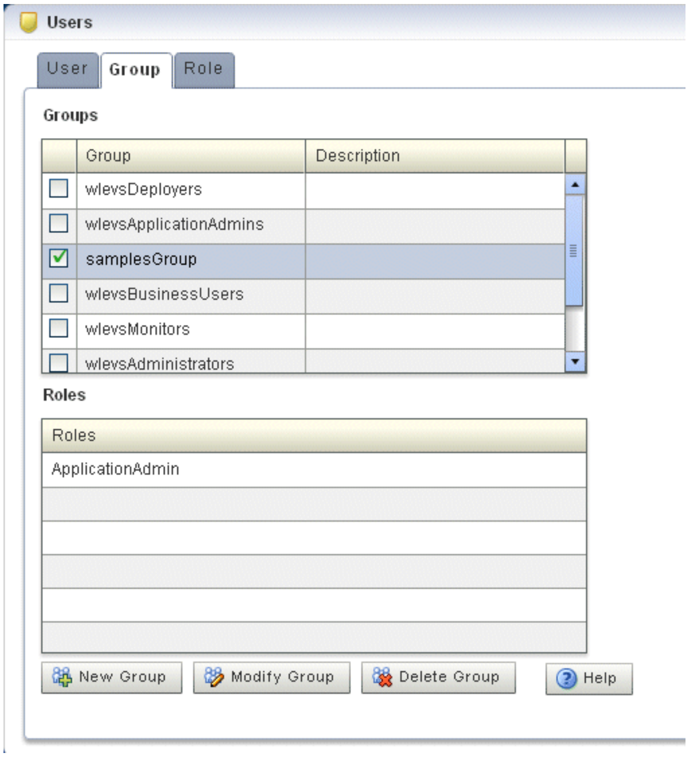
21.2 Change Group Roles

You can change existing groups, including the default groups, to change the roles they map to.

Change Group Roles

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **Group** tab.
4. In the Group table, check the boxes to the left of the name of the groups that you want to modify as Figure 21-3 shows.

Figure 21-3 Selecting a Group



Groups

	Group	Description
□	wlevsDeployers	
□	wlevsApplicationAdmins	
☑	samplesGroup	
□	wlevsBusinessUsers	
□	wlevsMonitors	
□	wlevsAdministrators	

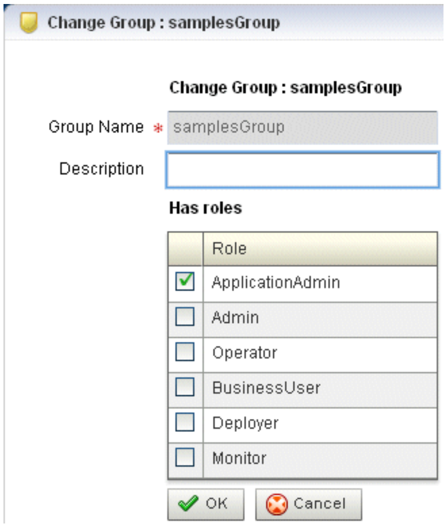
Roles

Roles
ApplicationAdmin

5. Click the **Modify Group** button at the bottom of the right pane.

   The Change Group panel appears as Figure 21-4 shows.

Figure 21-4 Change Group Panel



Change Group : samplesGroup

Group Name * samplesGroup

Description

Has roles

	Role
☑	ApplicationAdmin
□	Admin
□	Operator
□	BusinessUser
□	Deployer
□	Monitor

6. Configure the Change Group panel as Table 21-2 describes.

Table 21-2 Change Group Panel Attributes

Attribute	Description
`Group Name`	The name of the group. This is a read-only field..

Attribute	Description
`Description`	An optional description for this user.
`Has roles`	Check one or more roles to which the group maps. Each role grants access to an application. You must select at least one role. For more information, see Task Roles.

7. Click **OK**.

   When the group has been successfully modified, a confirmation message appears momentarily.

8. Optionally modify the description of the group.

9. Click OK.

21.3 Delete a Group

Oracle Event Processing is configured by default with a set of groups that are in turn mapped to roles: you cannot delete the default groups. See User, Group, and Role Management for details.

This section describes how to delete a group that you created.

Delete a Group

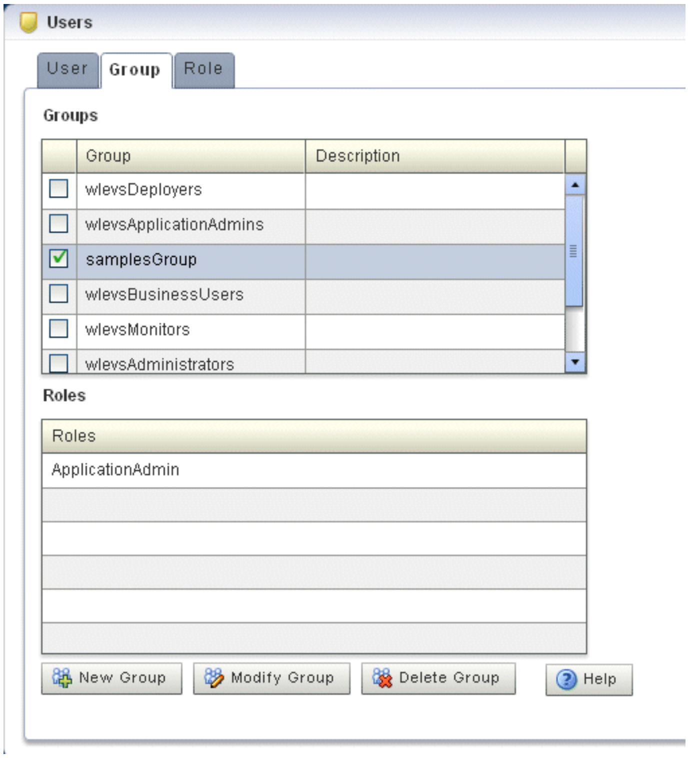
1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.

2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.

3. In the right pane, click the **Group** tab.

4. In the Group table, check the boxes to the left of the name of the groups that you want to delete as Figure 21-5 shows.

Figure 21-5 Selecting a Group



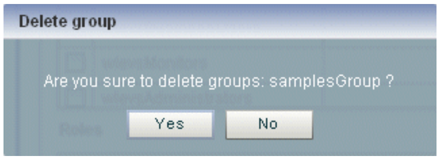
	Group	Description
□	wlevsDeployers	
□	wlevsApplicationAdmins	
☑	samplesGroup	
□	wlevsBusinessUsers	
□	wlevsMonitors	
□	wlevsAdministrators	

Roles
ApplicationAdmin

5. Click the **Delete Group** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 21-6 shows.

Figure 21-6 Delete Group Dialog



Are you sure to delete groups: samplesGroup ?

6. Click **Yes**.

   When the group has been successfully deleted, a confirmation message appears momentarily.

22

Task Roles

This chapter describes how you can use Oracle Event Processing Visualizer to create and delete roles that associate permissions with user groups.

This chapter includes the following sections:

- Create an Application Role
- Delete a Role
- Add the Application Role to a User

For more information, see User, Group, and Role Management.

22.1 Create an Application Role

Oracle Event Processing is configured by default with a set of task roles that are in turn mapped to groups. See User, Group, and Role Management for details.

This section describes how to create a new application role.

Create an Application Role

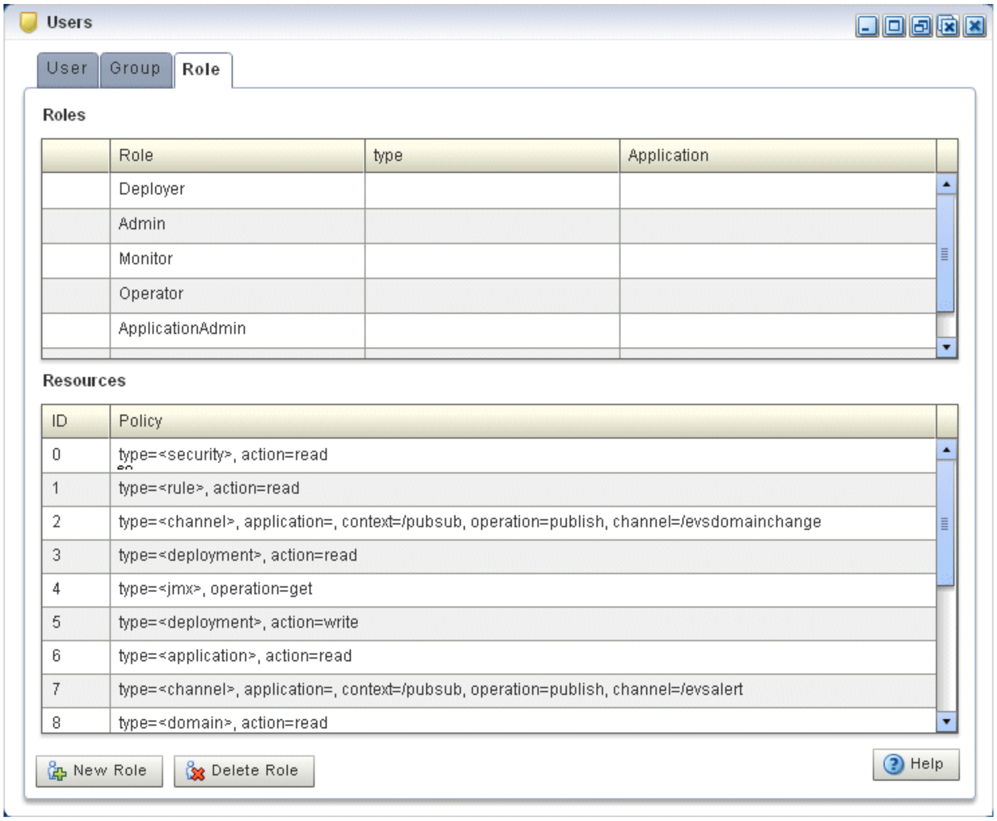
1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.

2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.

3. In the right pane, click the **Role** tab.

   The Role tab appears as Figure 22-1 shows.

Figure 22-1 Role Tab



Roles

Role	type	Application
Deployer		
Admin		
Monitor		
Operator		
ApplicationAdmin		

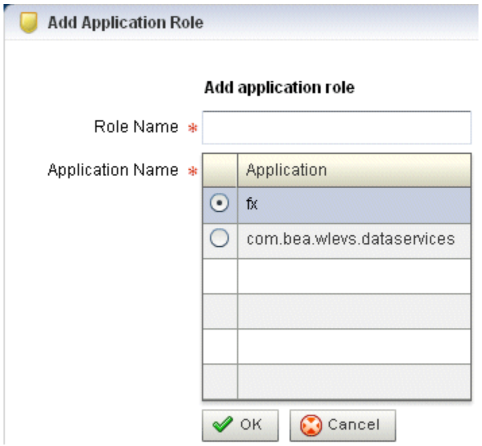
Resources

ID	Policy
0	type=<security>, action=read
1	type=<rule>, action=read
2	type=<channel>, application=, context=/pubsub, operation=publish, channel=/evsdomainchange
3	type=<deployment>, action=read
4	type=<jmx>, operation=get
5	type=<deployment>, action=write
6	type=<application>, action=read
7	type=<channel>, application=, context=/pubsub, operation=publish, channel=/evsalert
8	type=<domain>, action=read

4. Click the **New Role** button at the bottom of the right pane.

   The Add Application Role panel appears as Figure 22-2 shows.

Figure 22-2 Add Application Role



Add application role

Role Name *

Application Name *

	Application
●	fx
○	com.bea.wlevs.dataservices

5. Configure the Add Application Role panel as Table 22-1 describes.

Table 22-1 Add Application Role Panel Attributes

Attribute	Description
`Role Name`	Enter the name of the role.

Attribute	Description
`Application Name`	Select the radio button of the application you want to associate with this role from the list of all the currently running applications.

6. Click **OK**.

   When the application role has been successfully created, a confirmation message appears momentarily.

22.2 Delete a Role

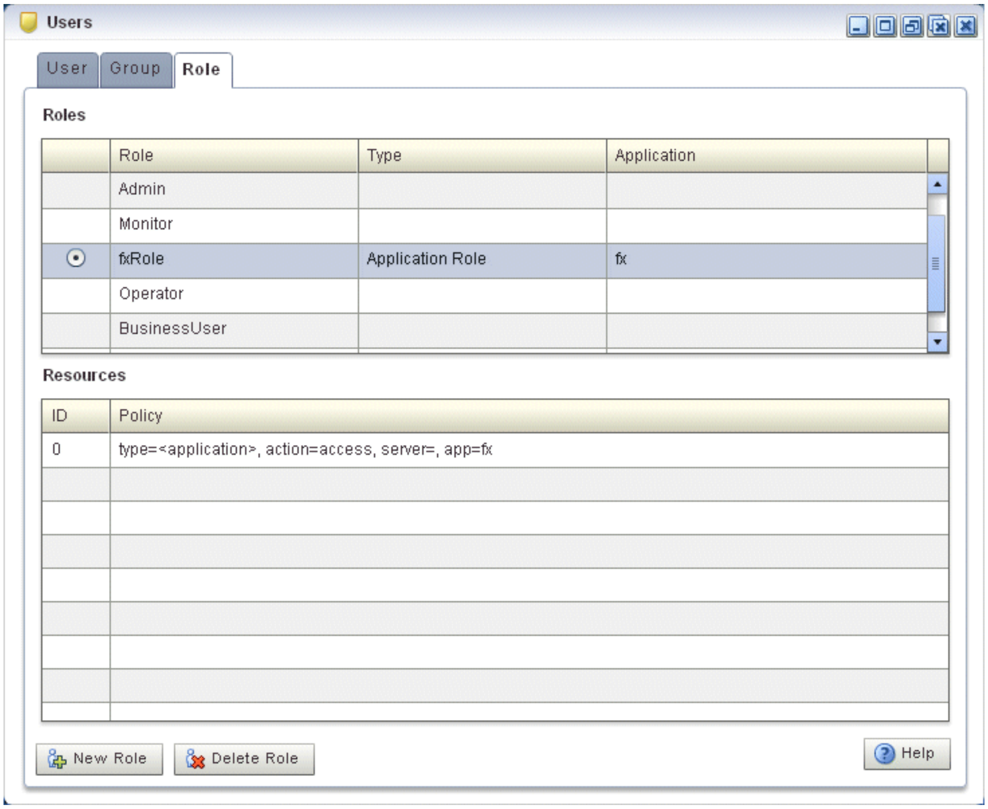
Oracle Event Processing is configured by default with a set of roles that are in turn mapped to groups: you cannot delete these default roles. See User, Group, and Role Management for details.

This section describes how to delete a role that you created.

Delete a Role

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > **Security** node, where ***Domain*** refers to the name of your Oracle Event Processing domain.
3. In the right pane, click the **Role** tab.
4. In the Role table, select the radio button to the left of the name of the role that you want to delete as Figure 22-3 shows.

Figure 22-3 Selecting a Role



Roles

	Role	Type	Application
	Admin		
	Monitor		
⊙	fxRole	Application Role	fx
	Operator		
	BusinessUser		

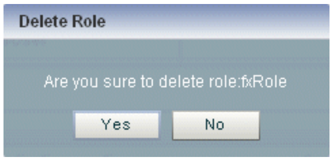
Resources

ID	Policy
0	type=<application>, action=access, server=, app=fx

5. Click the **Delete Role** button at the bottom of the right pane.

   A confirmation dialog appears as Figure 22-4 shows.

   Figure 22-4 Delete Role Dialog

   Delete Role

   Are you sure to delete role:fxRole

6. Click **Yes**.

   When the role has been successfully deleted, a confirmation message appears momentarily.

22.3 Add the Application Role to a User

For example, if user `testOEP` is a non-administrator who belongs to the `BusinessUser` group and needs access to application `fx`, you can use the following steps to set up user `testOEP` with the required access.

To Add the Application Role to a User to Grant Access to an Application

1. Create an application role as described in Create an Application Role with the following properties:

   Role Name: `fxRole`. Application Name: Select the `fx` radio button.

2. Create a group as described in Create a Group with the following properties:

   Group Name: fxGroup Has Roles: `fxRole`.

3. Edit the textOEP user as described in Edit a User with the following properties:

   Username: testOEP Belong to groups: `fxGroup`

23

HTTP Publish-Subscribe Server Security

You can use Oracle Event Processing Visualizer to specify roles that can publish to the HTTP publish-subscribe channels that are configured for the HTTP Publish-Subscribe server.

For more information, see HTTP Publish-Subscribe Server Channel Security Management.

Configure Security for an HTTP Publish-Subscribe Channel

1. Log on to Oracle Event Processing Visualizer as a user with the `Admin` role.
2. In the left pane, click the ***Domain*** > ***Server*** > Services > Http Pub/Sub Server node, where ***Domain*** refers to the name of your Oracle Event Processing domain and ***Server*** refers to the name of the server instance.

   A table appears in the right pane with the list of HTTP pub-sub servers configured for Oracle Event Processing.
3. In the right pane, click the name of the HTTP pub-sub server in the table. The default server is called `pubsub`.
4. In the Channels table, click the name of the channel for which you want to configure security.
5. In the Publish Roles table, select the roles that are allowed to publish messages to this channel.

   If you want to select more than one role, use the Ctrl key.
6. Click **Modify Channel** at the bottom of the pane.

24

SSL Configuration

You can view Oracle Event Processing server SSL configuration using the Oracle Event Processing Visualizer. To change the configuration, you must manually update the server's `config.xml` file.

For more information, see SSL Management.

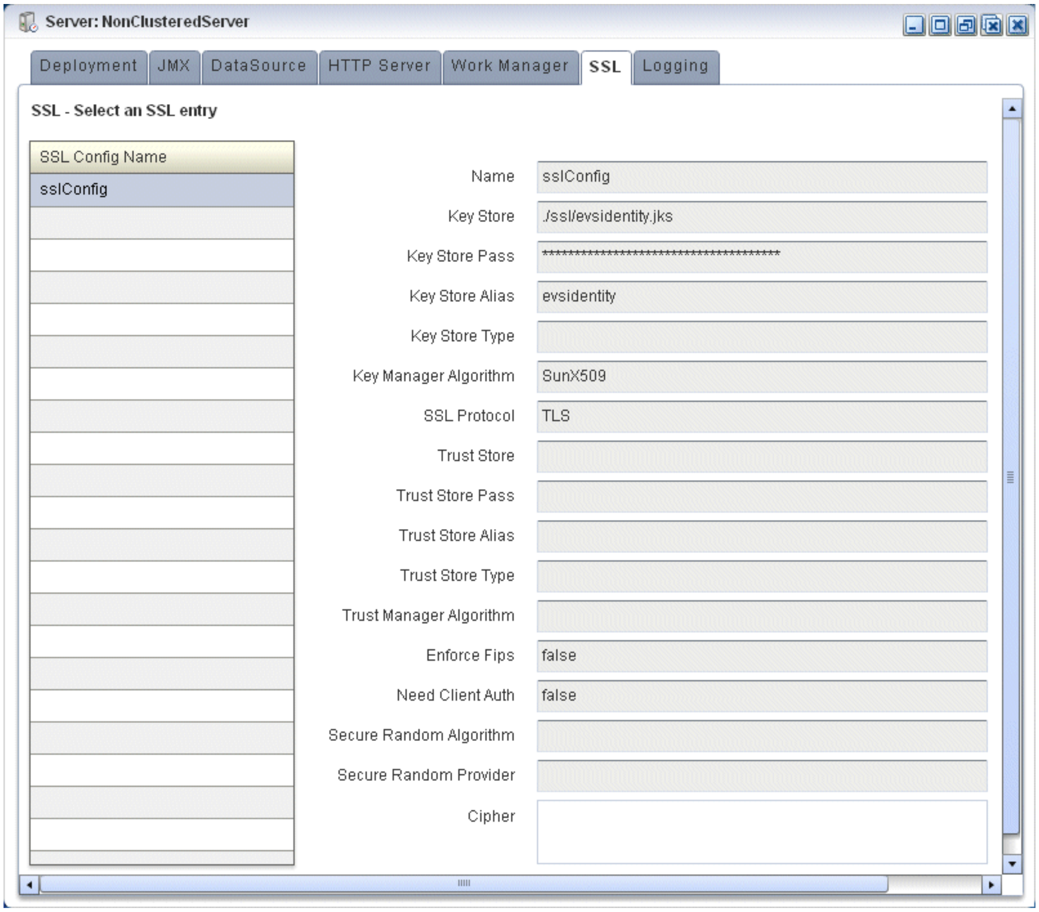
View the SSL Configuration for a Server

1. In the left pane, click the ***Domain*** > ***Server*** node, where ***Domain*** refers to the name of your Oracle Event Processing domain and ***Server*** refers to the name of the server instance.

2. In the right pane, click the **SSL** tab.

3. In the left table, click the SSL configuration you want to view as Figure 24-1 shows.

Figure 24-1 SSL Tab



SSL - Select an SSL entry

SSL Config Name
sslConfig

Name	sslConfig
Key Store	./ssl/evsidentity.jks
Key Store Pass	**
Key Store Alias	evsidentity
Key Store Type	
Key Manager Algorithm	SunX509
SSL Protocol	TLS
Trust Store	
Trust Store Pass	
Trust Store Alias	
Trust Store Type	
Trust Manager Algorithm	
Enforce Fips	false
Need Client Auth	false
Secure Random Algorithm	
Secure Random Provider	
Cipher	

The default configuration name is `sslConfig`.

4. View the SSL configuration options the right table displays.

   Table 24-1 lists the SSL configuration options.

Table 24-1 SSL Options

Option	Description
`Name`	The name of the selected SSL configuration.
`Key Store`	The file path and name of the key store certificate file contains a self-signed certificate. The file path is relative to the *`Oracle/Middleware/my_oep/user_projects/domains/DOMAIN_DIR/servername`* directory.
`Key Store Pass`	The key store password.
`Key Store Alias`	The key store alias.
`Key Store Type`	The key store type.
`Key Manager Algorithm`	The key manager algorithm.
`SSL Protocol`	The SSL protocol.

Option	Description
`Trust Store`	The file path and name of the key store certificate file contains a self-signed certificate. The file path is relative to the *`Oracle/Middleware/my_oep`*`/user_projects/domains/`*`DOMAIN_DIR/servername`* directory.
`Trust Store Pass`	The trust store password.
`Trust Store Alias`	The trust store alias.
`Trust Store Type`	The trust store type.
`Trust Manager Algorithm`	The trust store algorithm.
`Enforce Fips`	Whether or not Oracle Event Processing server uses a Federal Information Processing Standards (FIPS)-certified pseudo-random number generator for SSL.
`Need Client Auth`	Whether or not Oracle Event Processing server uses client authentication.
`Secure Random Algorithm`	The FIPS secure random algorithm, such as `FIPS186PRNG`.
`Secure Random Provider`	The FIPS secure random provider, such as `com.rsa.jsafe.provider.JsafeJCE`.
`Cipher`	The SSL ciphers.

A

Oracle CQL Query Wizard Template Schema Reference

This appendix includes the XSD for the schema behind Oracle Continuous Query Language (Oracle CQL) query templates.

The following example lists the XML schema document (XSD) you use when creating a template for use in the Oracle CQL Query Wizard.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.bea.com/ns/wlevs/config/queryconstructor"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.bea.com/ns/wlevs/config/queryconstructor">

    <!-- This is the first draft for the xsd used
        by the query constructor feature of the visualizer. -->


    <!-- The types of Operators supported by the query constructor-->
    <xs:simpleType name="operatorTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DStream" />
            <xs:enumeration value="Filter" />
            <xs:enumeration value="Intersect" />
            <xs:enumeration value="IStream" />
            <xs:enumeration value="Join" />
            <xs:enumeration value="Minus" />
            <xs:enumeration value="Output" />
            <xs:enumeration value="Pattern" />
            <xs:enumeration value="RStream" />
            <xs:enumeration value="Select" />
            <xs:enumeration value="Source" />
            <xs:enumeration value="Union" />
            <xs:enumeration value="Window" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="timeUnitTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="" />
            <xs:enumeration value="nanoseconds" />
            <xs:enumeration value="milliseconds" />
            <xs:enumeration value="microseconds" />
            <xs:enumeration value="seconds" />
            <xs:enumeration value="minutes" />
            <xs:enumeration value="hours" />
            <xs:enumeration value="days" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="windowTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="now" />
            <xs:enumeration value="range-time" />
            <xs:enumeration value="range-unbounded" />
            <xs:enumeration value="row-time" />
            <xs:enumeration value="rows" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="viewOutputTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Stream" />
            <xs:enumeration value="Relation" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="sourceTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="ExtRelation" />
            <xs:enumeration value="Stream" />
            <xs:enumeration value="Relation" />
            <xs:enumeration value="View" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="outputTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="View" />
            <xs:enumeration value="Query" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="patternSkipTypes">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DEFAULT" />
            <xs:enumeration value="ALL MATCHES" />
        </xs:restriction>
    </xs:simpleType>

    <!-- Outer element for all querys/views, root attribute points to the
        Output operator's ID for this query-->
    <xs:element name="Rule">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Operator" type="operatorType"
                    minOccurs="1" maxOccurs="unbounded" />
            </xs:sequence>
            <xs:attribute name="root" type="xs:integer" />
        </xs:complexType>
    </xs:element>

    <!-- Each "box" in the query constructor GUI is an Operator element -->
    <xs:complexType name="operatorType">
        <xs:sequence>

            <!-- Every operator has an associated CQL DDL -->
            <xs:element name="cql-property" type="xs:string"
                minOccurs="1" maxOccurs="1" />

            <!-- The inputs to this operator. A source operator has 0 inputs,
                all other operators have one or more inputs -->
            <xs:element name="inputs" type="inputsType" minOccurs="0"
                maxOccurs="1" />

            <!-- Some operators might have an alias, e.g. source, pattern -->
            <xs:element name="alias" type="xs:string" minOccurs="0"
                maxOccurs="1" />

            <!-- An operator may be any one of those defined in the operatorTypes enumeration -->
            <xs:choice minOccurs="1" maxOccurs="1">
                <xs:group ref="DStreamOperatorType" />
                <xs:group ref="FilterOperatorType" />
                <xs:group ref="IntersectOperatorType" />
                <xs:group ref="IStreamOperatorType" />
                <xs:group ref="JoinOperatorType" />
                <xs:group ref="MinusOperatorType" />
                <xs:group ref="OutputOperatorType" />
                <xs:group ref="PatternOperatorType" />
                <xs:group ref="RStreamOperatorType" />
                <xs:group ref="SelectOperatorType" />
                <xs:group ref="SourceOperatorType" />
                <xs:group ref="UnionOperatorType" />
                <xs:group ref="WindowOperatorType" />
            </xs:choice>
        </xs:sequence>

        <!-- What type of operator is this -->
        <xs:attribute name="type" type="operatorTypes" use="required" />

        <!-- This xml document is a tree of operators, the ID is used as a unique
            identifier to point to an operator.
            The input tag holds an ID of an operator which is an input to 'this' operator.
        -->
        <xs:attribute name="ID" type="xs:integer" use="required" />

    </xs:complexType>

    <!-- The inputs to an operator. Only a source operator has 0 inputs -->
    <xs:complexType name="inputsType">
        <xs:sequence>
            <xs:element name="input" type="xs:integer" minOccurs="1"
                maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <!-- DStream operator structure -->
    <xs:group name="DStreamOperatorType">
        <xs:sequence>
            <!-- there are no elements as of now in the DStream operator structure -->
        </xs:sequence>
    </xs:group>

    <!-- Filter operator structure -->
    <xs:group name="FilterOperatorType">
        <xs:sequence>
            <xs:element name="predicates" type="filterPredicatesType"
                minOccurs="1" maxOccurs="1" />
        </xs:sequence>
    </xs:group>

    <xs:complexType name="filterPredicatesType">
        <xs:sequence>
            <xs:element name="predicate" type="xs:string" minOccurs="1"
                maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>

    <!-- Intersect Operator structure -->
    <xs:group name="IntersectOperatorType">
        <xs:sequence>
            <!-- there are no elements as of now in the Intersect operator structure -->
        </xs:sequence>
    </xs:group>

    <!-- IStream operator structure -->
    <xs:group name="IStreamOperatorType">
        <xs:sequence>
            <!-- there are no elements as of now in the IStream operator structure -->
        </xs:sequence>
    </xs:group>

    <xs:group name="JoinOperatorType">
        <xs:sequence>
```

```
        <xs:element name="predicates" type="joinPredicatesType"
            minOccurs="1" maxOccurs="1" />
    </xs:sequence>
</xs:group>

<xs:complexType name="joinPredicatesType">
    <xs:sequence>
        <xs:element name="predicate" type="xs:string" minOccurs="1"
            maxOccurs="1" />
    </xs:sequence>
</xs:complexType>

<!-- Intersect Operator structure -->
<xs:group name="MinusOperatorType">
    <xs:sequence>
        <!-- there are no elements as of now in the Minus operator structure -->
    </xs:sequence>
</xs:group>

<!-- Output operator structure -->
<xs:group name="OutputOperatorType">
    <xs:sequence>
        <xs:element name="output-type" type="outputTypes" />
        <xs:element name="output-name" type="xs:string" />
        <xs:element name="view-schema-list" minOccurs="0"
            maxOccurs="1" />
    </xs:sequence>
</xs:group>

<xs:complexType name="viewSchemaList">
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
        <xs:element name="view-attr">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="name" type="xs:string" />
                    <xs:element name="type" type="xs:string" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>

<!-- Pattern operator structure -->
<xs:group name="PatternOperatorType">
    <xs:sequence>
        <xs:element name="pattern-skip" type="patternSkipTypes"
            minOccurs="1" maxOccurs="1" />

        <xs:element name="partition-by" minOccurs="0"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="partition-attr"
                        type="xs:string" minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="define-list" minOccurs="1"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="define-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="measures-list" minOccurs="1"
```

```
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="measure-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="pattern-list" minOccurs="1"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="pattern-attr" type="xs:string"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="subsets" minOccurs="0" maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="subset" type="subsetType"
                        minOccurs="1" maxOccurs="unbounded" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        <xs:element name="timer-event" minOccurs="0"
            maxOccurs="1">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="duration" type="xs:string" />
                    <xs:element name="multiple-duration"
                        type="xs:boolean" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:group>

<xs:complexType name="subsetType">
    <xs:sequence>
        <xs:element name="subset-name" type="xs:string" />
        <xs:element name="corr-attr-name" type="xs:string" />
    </xs:sequence>
</xs:complexType>

<!-- Rstream operator structure -->
<xs:group name="RStreamOperatorType">
    <xs:sequence>
        <!-- there are no elements as of now in the RStream operator structure -->
    </xs:sequence>
</xs:group>

<!-- Select operator structure -->
<xs:group name="SelectOperatorType">
    <xs:sequence>
        <xs:element name="select-list" type="selectListType"
            minOccurs="1" maxOccurs="1" />

        <xs:element name="group-list" type="groupListType"
            minOccurs="0" maxOccurs="1" />

        <xs:element name="having-clause" type="xs:string"
            minOccurs="0" maxOccurs="1" />

        <xs:element name="order-by-list" minOccurs="0"
            maxOccurs="1" />
    </xs:sequence>
</xs:group>


<xs:complexType name="groupListType">
    <xs:sequence>
        <xs:element name="group-by" type="xs:string" minOccurs="1"
            maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>


<xs:complexType name="orderByListType">
    <xs:sequence>
        <xs:element name="order-by-attr" type="orderByAttrType"
            minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="orderByAttrType">
    <xs:sequence>
        <xs:element name="order-by-symbol" type="xs:string" />
        <xs:element name="ascending" type="xs:boolean" />
        <xs:element name="nulls-first" type="xs:boolean" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="selectListType">
    <xs:sequence>
        <xs:element name="select-attr" type="selectAttrType"
            minOccurs="1" maxOccurs="unbounded">

        </xs:element>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="selectAttrType">
    <xs:sequence>
        <xs:element name="select-expression" type="xs:string" />
        <xs:element name="alias" type="xs:string" />
    </xs:sequence>
</xs:complexType>

<!-- Source operator structure -->
<xs:group name="SourceOperatorType">
    <xs:sequence>
        <!-- The name of this source in the CQL engine -->
        <xs:element name="source-name" type="xs:string" />

        <!-- Is this a 'View' / 'Stream' / 'Relation' -->
        <xs:element name="source-type" type="sourceTypes" />

        <!-- If this is a view, does it produce a 'Stream' or a 'Relation' -->
        <xs:element name="view-output-type" type="viewOutputTypes"
            minOccurs="0" maxOccurs="1" />
    </xs:sequence>
</xs:group>

<!-- Intersect Operator structure -->
<xs:group name="UnionOperatorType">
    <xs:sequence>
        <!-- Is this a union all statement? true/false -->
        <xs:element name="union-all" type="xs:boolean" />
    </xs:sequence>
</xs:group>

<!-- Window operator structure -->
<xs:group name="WindowOperatorType">
    <xs:sequence>
```

```
            <!-- what type of window is this -->
            <xs:element name="type" type="windowTypes" />

            <!-- only for partition windows -->
            <xs:element name="partition-by" type="xs:string"
                minOccurs="0" maxOccurs="1" />

            <!-- required for all windows except "now" -->
            <xs:element name="range-params" type="windowRangeParamsType"
                minOccurs="0" maxOccurs="1" />
        </xs:sequence>
    </xs:group>

    <xs:complexType name="windowRangeParamsType">
        <xs:sequence>
            <xs:element name="timeSpecType" minOccurs="0" maxOccurs="1" />
            <xs:element name="slideSpecType" minOccurs="0"
                maxOccurs="1" />
            <xs:element name="rows" type="xs:integer" minOccurs="0"
                maxOccurs="1" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="slideSpecType">
        <xs:sequence>
            <xs:element name="slidevalue" type="xs:integer" />
            <xs:element name="slideunit" type="timeUnitTypes" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="timeSpecType">
        <xs:sequence>
            <xs:element name="timevalue" type="xs:decimal" />
            <xs:element name="timeunit" type="timeUnitTypes" />
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```