Cisco Prime Analytics 1.0 User Guide

July 26, 2013

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

Text Part Number: OL-28757-01

Cisco Prime Analytics 1.0 User Guide

CONTENTS

Preface

Cisco Prime Analytics is a scalable, real-time analytics platform that allows for adaptation to customer environments and use cases. This guide describes the architecture, supporting hardware and software, and management procedures for Prime Analytics. The information in this guide helps you to complete the tasks that are necessary to use Prime Analytics to monitor the performance of your network.

This section describes the audience, organization, and conventions of the *Cisco Prime Analytics 1.0 User Guide*. It refers you to related publications and describes online sources of technical information.

For a more detailed description of Prime Analytics, see Chapter 1, "Prime Analytics Overview." For the latest Prime Analytics information and software updates, go to http://www.cisco.com/go/analytics.

This preface includes the following topics:

Audience

This guide is for system administrators, network operators and basic users who use Prime Analytics for reporting on the managed network. They should have:

- Basic network management skills
- Basic Linux system administrator skills

Organization

This guide is divided into the following chapters and appendices:

- Chapter 1, "Prime Analytics Overview" provides brief descriptions of Prime Analytics architecture, and an overview of how to use Prime Analytics.
- Chapter 2, "Setting Up Prime Analytics Projects" tells you how to set up a Prime Analytics project.
- Chapter 3, "Navigating the Prime Analytics User Console" tells you how to log into Prime Analytics and provides an overview to the Prime Analytics GUI.

- Chapter 4, “Setting Up the Data Sources” tells you how to set up the data sources in the BI platform.
- Chapter 5, “Creating Dashboards” tells you how to use the Prime Analytics BI platform User Console to create dashboards.
- Chapter 6, “Creating Reports” tells you how to use the Prime Analytics BI platform User Console to create reports.
- Chapter 7, “Creating an Analysis” tells you how to use the Prime Analytics BI platform User Console to create an analysis.
- Chapter 8, “Managing Prime Analytics” provides all Prime Analytics management procedures including management of users, backups, and many other administrative tasks.
- Appendix A, “Real Time Chart Parameters” provides a list of parameters used in the real time charts.
- Appendix B, “Sample Applications” provides a descriptions of the sample applications provided with Prime Analytics.
- Appendix C, “Advanced Topics” provides additional topics that provide more in-depth understanding of Prime Analytics.

Conventions

This document uses the following conventions:

Item	Convention
Commands and keywords	**boldface** font
Variables for which you supply values	*italic* font
Displayed session and system information	`screen` font
Information you enter	**`boldface screen`** font
Variables you enter	*`italic screen`* font
Menu items and button names	**boldface** font
Selecting a menu item in paragraphs	**Option > Network Preferences**
Selecting a menu item in tables	Option > Network Preferences

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the publication.

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Tip Means *the following is a useful tip*.

Product Documentation

You can access the following additional Cisco Prime Analytics guides on the Cisco Prime Analytics page on Cisco.com:

- *Cisco Prime Analytics 1.0 User Guide* (this guide)
- *Cisco Prime Analytics 1.0 Release Notes*
- *Cisco Prime Performance Manager 1.0 Quick Start Guide*
- *Open Source Used in Cisco Prime Analytics 1.0*
- *Cisco Prime Analytics 1.0 Documentation Overview*

Prime Analytics data sheet can be found at http://www.cisco.com/go/performance

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly What's New in Cisco Product Documentation, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

Subscribe to the What's New in Cisco Product Documentation as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS Version 2.0.

Prime Analytics Overview

Cisco Prime Analytics is a real-time big data analytics platform that allows you to connect to continuos streams of NetFlow, syslog, and XMPP data and run queries on both live and stored data to answer real time business questions.

The following topics provide an overview to Cisco Prime Analytics features and architecture:

Features and Functions

Prime Analytics provides the following features and functions:

- Streaming data—Prime Analytics can process large streaming data volumes, which allows decisions to be made based upon current data trends and analysis.
- Continuous queries—To interpret continuous data streams, continuous queries are required. Prime Analytics continuous queries are always on. New results are generated whenever new data arrives. The data is routed through the active query set and new results are published to downstream subscribers.
- Windows—Prime Analytics windows allow continuous data streams to be divided into segments so that results can be presented at meaningful points. Windows can be based on time intervals or by the number of arriving records.
- Streaming views—Prime Analytics allows you to create views of streaming data for higher-level real-time data analysis.
- SQL support—Prime Analytics supports SQL for real-time data stream queries. Supported SQL functions include stored procedures, user-defined functions, user-defined aggregates, joins, and other functions.
- Replay and drill-down—To learn from a continuous event stream and improve organization responses to them, Prime Analytics allows you to replay, analyze, and drill into data. This allows you to evaluate and tune organization event responses.
- Shared processing—To handle continuous query performance demands, Prime Analytics uses parallel processing to maximum advantage and ensure performance degradation does not occur.
- Adaptability—The Prime Analytics continuous query engine is adaptive; it can accommodate dynamic additions, removals, or modification of queries on-the-fly, without requiring a system restart.

- Support for data streams and data tables—Prime Analytics allows you to run continuous queries over data streams and data tables.

Architecture Overview

Prime Analytics components include a continuous data analysis engine that handles both real-time and historical data, a data integration layer that moves data into and out of the system, and a visualization layer that provides dynamic dashboards and reports to end users.

Prime Analytics components include the continuous query (CQ) engine, integration framework, and business intelligence (BI) platform.

CQ Engine

The Prime Analytics continuous query engine, called TruCQ performs all continued and historical data processing functions and stores results in the PostgresSQL Prime Analytics database.

Integration Framework

The Prime Analytics data integration framework is a Java platform that allows you to configure connectors and handlers to external data sources and data targets. The integration framework customizes communication with the CQ engine at the data input and output points. Integration framework connectors handle data transport protocol integration. Connectors are drivers that connect to external applications and support the input of the streaming, batch, structured, and unstructured data types. Connectors also handle data transformation functions as required by the query engine or the target application, for example, reformatting dates or combining a device ID with static data contained in an external database.

Any given data integration point involves multiple aspects including transport, format, schema, and others. With data input, for example, transport governs how the data arrives at the integration framework and is generally associated with a wire protocol such as HTTP, NetFlow, syslog, and XMPP. Format governs the message payload. It can take various forms such as delimited text, XML, JSON, Google Protocol Buffers, Apache Thrift, and others. Finally schema governs the integration point output. It is a structured format suited for pushing data into the continuous query database.

To handle the transport, format, and schema requirements, the Prime Analytics integration framework offers two mechanisms: connectors and handler chains. A connector is a single monolithic Java component that is responsible for all data integration point areas. You could, therefore, have a single component that extracts XML data from a JMS queue and massages the XML payload according to an Extensible Stylesheet Language stylesheet to render the output as a tab-delimited structured schema.

In contrast, the handler chain modularizes the various connector elements into independent handlers that can in turn be chained together to solve a given data integration problem. The handler chain could, therefore, have separate handlers for extracting arbitrary objects from a JMS queue and for converting an XML object using an XSLT stylesheet. The primary advantage of the handler chain is reusability. If, for example, you have a new data integration point that sends JSON data over a JMS queue, or XML data over an XMPP transport, you simply need to implement a single new handler for each case.

Connectors and handler chains are both supported, However, the handler approach is recommended. An additional advantage of the handler chain is the ability to take a user-implemented handler chain and turn on multi-threading.

BI Platform

The Prime Analytics BI platform is built from third party open source components including Pentaho, Saiku, and others. The BI provides the visual representation for continuous query data and provides tools you can use to create dashboards and reports that present data in ways suited to your specific business needs.

Continuous Query Engine

The Prime Analytics continuous query engine is the foundation of the Prime Analytics architecture. The query engine performs all data processing and analysis functions, including standard relational database operations as well as continuous queries against continuous streams of data. A data stream is an unbounded, potentially infinite, series of records, or tuples, traveling through a network. Similar to data tables, a stream is a database object with an associated structured schema. Data stream examples include:

- Flow events from network routers and switches.
- Video player logs.
- Network event or security logs.
- Website click and impression stream data.
- Sales data from distributed point-of-sale terminals.
- Data feeds from sensors, barcodes, or radio frequency identifiers (RFIDs).
- Financial tick data from securities exchanges.
- Transaction data over debit or credit card networks.
- Service-oriented architecture (SOA) components on an enterprise message bus.

A data stream is any series of tuples that grow over time. The decision to categorize and manage these data flows as streams instead of writing them to database tables is based on the data volume, sequence, and the latency requirements of the applications that depend on the data.

Prime Analytics streams can be one of the following types, depending how the stream is populated:

- Raw streams—Are populated by an external data provider that connects to Prime Analytics using a well-defined protocol and pumps in data for the Prime Analytics to use.
- Derived streams—Are defined using a query, called a defining query, and populated by the continuous query engine. A derived stream can be one of two types:
  - View—Similar to a regular database view, a streaming view creates a virtual stream that can be used by other queries instead of a raw stream. A view's defining query only runs when a query using the view is running.
  - Persistent continuous query—Is a materialized continuous query explicitly associated with a stream. It is similar to a view, but does not have macro semantics and is always active.

Each stream record that enters Prime Analytics has a time stamp attribute. (This attribute is also called cqtime.) The time stamp attribute is either provided by the source or user, or is system-generated. The user-provided time stamp is preferred when the original event time stamp is required to establish a continuous query window. The system-generated time stamp is preferred when the original time stamp is not important.

Windows

Prime Analytics uses windows to divide the continuous data streams into discrete data sets. Certain operations, such as aggregation, require finite data sets to generate results. Windows may be time-based or record-based, depending on the query requirements. Windows are computed using time-stamp attributes for time-based streams and internal record counters for record-based windows. The window defines a snapshot for a given data stream to create a finite set of tuples on which queries can produce results.

Continuous Queries

Continuous data stream queries run continually and concurrently. They produce results in continuous, zero-latency, output streams. Prime Analytics queries follow SQL query syntax. Any SQL queries and data processing models created for traditional database systems can be used in Prime Analytics. Supported SQL query functions include:

- Views.
- Filters, for example, such operators as >, <, and =.
- User-defined functions, for example, C, C++, Java, and Perl.
- Joins over streams and relations
- Aggregates, for example, SUM, COUNT, MIN, MAX, and AVG.
- User-defined aggregates.
- Grouped aggregates.
- Arbitrary subselects.
- Other subqueries

The continuous query engine performs queries directly against the data streams. It does not need to persist data to generate results. However, you can archive queries so they are persisted to the analytics tables for later access, replay, drill down, analysis, data enrichment, or historical correlations.

Replay and Drill Down

During an ongoing flow of continuous stream queries, you sometimes might want to review an event sequence again to fully understand what happened. When this occurs, Prime Analytics allows you to pause and replay the stream to review the event sequence. You can also take a data snapshot and drill into it to analyze the event and the system response. This replay and drill down capability enables you to better understand how to respond to business events and tune responses over time. You can take a snapshot of data from a particular time sequence, replay the stream, then create drill-down reports to understand and learn from the event sequence.

Prime Analytics Stream Relational Database Management System

The Prime Analytics continuous query database is a relational database extended for doing big data analytics. The continuous query database is based on the PostgreSQL open source object-relational database system. PostgreSQL functionality is available for continuous query data including tables, caching, archiving, replay, queries over streams. PostgreSQL modules packaged with Prime Analytics include the postmaster, system catalog, query parser, query optimizer, ODBC and JDBC adapters, user-defined functions and aggregates, as well as plugins for external programming languages.

Integration Framework

The Prime Analytics data integration framework is a Java platform that allows you to configure connectors to external data sources and data targets. The integration framework communicates with the continuous query engine at the data input and output points over a unified data integration layer. Integration framework connectors handle data transport protocol integration. Connectors are software components that connect to external applications. Connectors support the input and output of streaming and batch data sets. The data sets can be naturally structured or unstructured. Connectors also handle data transformation functions as required by the query engine or the target application, for example, reformatting dates or combining device ID with static data contained in an external database.

Connectors can be used for either input or output of data. Input connectors, also called producers, integrate external data sources that send data streams to the query engine. These include message queues, enterprise buses, proprietary APIs, and other data sources. Input connectors support the following source types:

- Pull sources—Used for traditional database systems.
- Push sources—Initiated by the connector (push-client) or by the data source (push-server).

Output connectors, also called consumers, deliver query results from the query engine to external source systems. Continuous streaming queries produce data streams. Like input connectors, output connectors can support different modalities depending on the client interaction:

- Push targets—Used for clients that can accept continually streamed query results.
- Pull targets—Used for clients that support only intermittent result retrievals.

Prime Analytics provides a metadata catalog of common connectors that cover most major data formats including:

- Syslog
- NetFlow
- XMPP
- CSV
- JDBC
- ODBC
- JMS
- Log4J
- File
- Retry
- Throttle

Connectors are created using an XML template. You can modify connectors or create new ones using an XML editor.

Business Intelligence Platform

The Prime Analytics business intelligence platform consists of open source applications developed by Pentaho, Saiku, and others, to create a web-based, client that can generate dynamic dashboards and reports from continuous query data. The BI platform includes functions for integration, information

delivery, and basic analysis. Information delivery elements include reports, dashboards, and ad hoc queries. Basic analysis elements include Online Analytical Processing (OLAP) and interactive visualizations.

Security

Prime Analytics security functions include:

- Role-based password-protected access for multiple users
- Multiple user authentication methods (PAM-based and standalone)
- Web-based and CLI-based user management
- Password enforcement policies (aging, minimum length, and lockouts)
- Audit trails of all user actions and all access through the web interface
- Security logs

In addition, you can enable SSL for added security between client web browsers and the Prime Analytics BI platform.

Setting Up Prime Analytics Projects

After you install Prime Analytics, you must complete the following procedures to set up Prime Analytics projects. These include setup of handlers and connectors that link the data source to the CQ engine, and CQ engine output to the BI platform. Additionally, you must define queries to display the data in which you are interested.

Topics include:

Creating the Project

Creating a project creates a dedicated project directory where all application elements are stored.

To create the Prime Analytics project:

Step 1 Verify the path to the Java Runtime Environment (JRE) is set correctly:

```
source /etc/sysconfig/primeanalytics/primea
export JAVA_HOME
```

Step 2 Verify the TRUVISO_HOME variable path.

TRUVISO_HOME is an environment variable defined in a script called /etc/sysconfig/primeanalytics/primea. The script contains other variables such as JAVA_HOME and PA_HOME. To build or run applications, you must define all environment variables correctly. The easiest way to define them is to run the script, $PA_HOME/bin/pa_env.sh, where PA_HOME is set by /etc/sysconfig/primeanalytics/primea. You must source $PA_HOME/bin/pa_env.sh to set environment variables.

Step 3 Verify that your home directory has a location for your projects:

```
ls -l ~/projects
```

If the projects directory is not present, create one:

```
mkdir ~/projects
```

Step 4 Create the application:

```
cd ~/projects
$TRUVISO_HOME/Scripts/make-application.sh projectname
cd ~/projects/projectname
ls -l
```

Note If you see an invalid project name warning, retype the original project name.

Your project directory should now contain a customization directory. It contains files that you can edit, including:

- build-local.sh—Calls build.xml to compile the Java and custom modifications.
- build.xml—Contains project parameters.
- start.sh—Starts the application.
- stop.sh—Stops the application.

Defining the Application and Database Streams

After you create your project, the next step is to define the continuous stream application and its database streams. To accomplish this, you edit the customizations/db/ddl.sql file.

To define the continuous stream application and its database streams:

- From root project directory, apply the application DDL.

  ```
  ant run-ddl
  ```

  or

  ```
  psql -U primea -a -f customizations/db/ddl.sql
  ```

Verify the DDL

Now you will verify the DDL.

To verify the DDL:

Step 1 Run the following command. You can run it from any directory as long as TRUVISO_HOME/TruCQ/bin is added to the PATH.

```
psql -C
```

Step 2 Set the search path to show the created schemas, streams, and tables for the application and Prime Analytics object model schema (truviso_metadata):

```
set search_path=my_schema,public;
```

Step 3 Exit the PostgreSQL terminal:

```
\q
```

Managing Connectors and Handlers

Prime Analytics connectors and handlers connect to data sources and populate data streams. Connectors extract and transform the data and integrate it with the CQ engine. They also integrate CQ engine output with enterprise applications. Connectors provide the basic infrastructure to manage the CQ life cycle. To fine tune the data processing, handlers are provided to manage the small streaming data details including input, record parsing, and data flow management coming in and out of the CQ engine.

Connectors are deployed in the following directories:

- Connectors—*Project*/customizations/src/com/truviso/cq/connector
- Handlers—*Project*/customizations/src/com/truviso/system/handlers

To configure connectors and handlers:

Step 1 Review the data.xml file located in the MyProject/customzations/templates/.

The data.xml file is the root configuration file for a Prime Analytics project. All customizations can be performed by editing this file. However, the best practice is to create a pointer to a separate file for maintenance and support cases.

Step 2 Verify the following exists in the connector data.xml file in the <!DOCTYPE> section:

```
<!ENTITY connectors SYSTEM "include-connectors.xml">
```

Step 3 Verify the following appears within another tag anywhere in the <config> section:

```
&connectors;
```

This entry allows you to configure all connectors and/or handlers in the include-connectors.xml file instead of the data.xml file. The include-connectors.xml file should be present in the same directory as the data.xml file.

Step 4 Edit the include-connectors.xml file to include the parameters and definitions required by the connectors and/or handlers.

Step 5 Proceed to one of the following procedures, depending upon the data type that will generate the continuous queries for your project:

- Managing Handler Chains, page 2-4
- Building a NetFlow Application, page 2-8
- Building a Syslog Application, page 2-12
- Building an XMPP Connector, page 2-19

Managing Handler Chains

Handlers manage the chain of events required to push data into the Prime Analytics CQ engine. The <producers> element contains a list of <chain> elements. Each element defines a single handler chain. Prime Analytics built-in data handlers are listed in Table 2-1.

Table 2-1 Prime Analytics Handlers

Handler	Description
file	Opens a file and return an InputStream.
inputstream	Reads a line off text an InputStream.
objectstream	Converts InputStream into ObjectInputStream.
logging	Logs the input using log4j.
socket	Listens on a socket and produce InputStream.
loggingevent	Handles an log4j LoggingEvent.
text	Handles and parse a text string.
throttle	Varies the data rate of a text-based record containing timestamp, one of which is the CQTIME column of the destination stream.
pump	Pumps input to a CQ Engine.
jms	A JMS queue consumer. Uses standard JMS parameters.
message	Handles a JMS Message.
custom	Defines a custom handler.
commit	Commits data at regular intervals.
retry	Attempts to push uncommited data again into engine in case of failure.
twophasepump	Pushes data into engine with multiple threads with 2-phase commit semantics.

Table 2-3 shows the handlers organized by function.

Table 2-2 Handlers by Function

Function	Handlers
Transport	file, socket, udp
Transform	inputstream, objectstream, logging, loggingevent, text, xml
Testing	throttle
Save data to the database	pump, commit, retry, twophasepump
Handle syslog messages	udpsyslog, syslog
Handles NetFlow messages	udpnetflow, netflow
Handles XMPP messages	xmpp, xml
Customization	custom

Table 2-3 Prime Analytics Handlers

Handler	Description
file	Opens a file and return an InputStream.
inputstream	Reads a line off text an InputStream.
objectstream	Converts InputStream into ObjectInputStream.
logging	Logs the input using log4j.
socket	Listens on a socket and produce InputStream.
loggingevent	Handles an log4j LoggingEvent.
text	Handles and parse a text string.
throttle	Varies the data rate of a text-based record containing timestamp, one of which is the CQTIME column of the destination stream.
pump	Pumps input to a CQ Engine.
jms	A JMS queue consumer. Uses standard JMS parameters.
message	Handles a JMS Message.
custom	Defines a custom handler.
commit	Commits data at regular intervals.
retry	Attempts to push uncommited data again into engine in case of failure.
twophasepump	Pushes data into engine with multiple threads with 2-phase commit semantics.

You can create parallel handler chains to optimize performance. Parallel chains can include built-in handlers and any custom handlers you create. To enable multi-threaded data processing, set the handler queue attribute to indicate the handler runs in parallel mode. Additional threads and queue size attributes allow you to fine tune the process. All down stream processing is performed in the threads initiated for this handler.

If you use multiplexing and full transactional semantics for publishing data to the continuous query engine, use the twophasepump handler instead of the regular pump handler to push the data. This ensures that all data is aborted or committed. The twophasepump handler uses the PostgreSQL prepared transactions feature. If you use the 'twophasepump, raise the max_prepared_transactions setting in postgresql.conf. The default is 5. This is sufficient for a typical installations, but is not sufficient if you use the twophasepump handler extensively. The recommended practice is to set this to the same value as max_connections.

If you use prepared transactions, monitor the pg_catalog.pg_prepared_xacts table periodically for stranded prepared transactions. For a full description of the prepared transaction feature, consult the PostgreSQL documentation.

Available parameters include:

queue—Enables the queue-based, multi-threaded capabilities for the current handler. The default is false

- threads—Specifies the number of threads to use to process data passed to the handler. The default is 10.
- queuesize—Specifies size of the queue to use to transfer data to threads processing data. The default is 1000.

Defining the Handler

To enable the Prime Analytics BI platform to recognize handler, you must define it in either the include-connectors.xml or data.xml configuration file. Required configuration parameters to include in the <handler> xml definition include:

- FILE —Opens a file and returns an InputStream.

  ```
  <file> <location>/path_to_file/file</location> </file>
  ```

  The location parameter is required by the handler in order to proceed. In this case, it requires the location of the file that needs to be processed.

- SOCKET—Creates a socket on a specific port and listens for incoming connections.

  ```
  <socket> <port>1234</port> </socket>
  ```

  THROTTLE—Modifies the incoming data rate. This handler is often used to simulate high peak traffic periods for performance testing. The parameters are:

  - cqtimecolumnindex—The column index of the USER CQTIME column.
  - pumpratio—Ratio to speed up the data replay.
  - cqtimedatepattern—The timestamp format of values in the CQTIME column.
  - keepOriginalCqTime—If true, retains the incoming CQTIME value. If false, the incoming CQTIME value is replaced with the current system time.

  ```
  <throttle> <cqtimedatepattern>yyy-MM-dd HH:mm:ss</cqtimedatepattern>
  <cqtimecolumnindex>1</cqtimecolumnindex> <pumpratio>10</pumpratio>
  </thottle>
  ```

- JMS—Consumes messages off a specific Java Message Service (JMS) queue.

  ```
  <jms> <jmsBrokerId>broker1</jmsBrokerId>
  <jmsUri>jms:queue:ExampleQueueNamefitimeToLive=1000</jmsUri>
  <jmsContextFactory>org.apache.activemq.jndi.ActiveMQInitialContextFactory</jmsContextF
  actory> <jmsConnectionFactory>QueueConnectionFactory</jmsConnectionFactory>
  <jmsQueue>queue1</jmsQueue> <jmsUsername>jms_user</jmsQueue>
  <jmsPassword>jms_pwd</jmsPassword>
  </jms>
  ```

- COMMIT—Issues a commit every given number of rows.

  ```
  <commit> <rows>1000</rows> </commit>
  ```

  Although transaction size depends on your application requirements, as a general guideline, set commits to one per second for high data rates. For example, for a NetFlow with 50,000 flows (rows) per second, you would set commit to 50000 rows. You can set faster and slower commit rates. However, never set commit rates less than 100 per second to avoid problems caused by the volume of overhead operations for each commit.

Handler Chain Example

The following sample handler chain pulls data from a comma delimited sample_file.txt file. The file is a small representation of the actual source data. To simulate a production environment, the file is replayed and throttled to simulate large data volumes and fast data rates. The file and throttle handlers are used in this example. The file handler is extended to include replay capability as follows:

```
import com.truviso.system.handlers.* ;
public class FileReplayHandler extends AbstractHandler<Connector, InputStream> { File
```

```
    file;
private static final Logger LOGGER = Logger.getLogger(FileReplayHandler.class);
private static final int DEFAULT_MAX_CACHE_SIZE = 5000;
private ArrayList<Object> inputCacheList; private int iterations; private int
    maxcachesize;
private String location;
public FileReplayHandler(SubnodeConfiguration c) throws HandlerException { super(c); }
public void init() throws HandlerException { super.init();

iterations = config.getInt("iterations"); maxcachesize = config.getInt("maxcachesize");
location = config.getString("location"); if (location == null)
    throw new HandlerException("Please specify location element for file handler");
file = new File(location); if (!file.exists())
    throw new HandlerException(String.format("File %s does not exist", location));
maxcachesize = maxcachesize<=0fiDEFAULT_MAX_CACHE_SIZE:maxcachesize; inputCacheList =
    new ArrayList<Object>();
}
public void handle(Connector producer) throws HandlerException {

for (int i = 0; i < iterations && isRunning(); i++)
{
LOGGER.debug("================================================================================
================" LOGGER.debug("iteration: " + i);
LOGGER.debug("================================================================================
================" try {
InputStream in = new FileInputStream(file); resultListener.handle(in); } catch
(FileNotFoundException e) { throw new HandlerException(String.format("Unable to read file
%s", file.getAbsolutePath }
} }
}
```

The following parameters are added:

- iterations—The number of times the file will be replayed.
- maxcachesize—The maximum amount of memory utilized for the records in the file allocated to cache. The handler chain configuration is defined below:

```
<handlers> <chain id="FILE_REPLAY_THROTTLE">
<custom class="com.truviso.system.handlers.FileReplayHandler">
<iterations>1000</iterations> <location>/tmp/source_file.txt</location>
<maxcachesize>5000</maxcachesize>
</custom>
<inputstream/>
<text> <delimiter>,</delimiter>
</text>
<custom class="com.truviso.system.handlers.ThrottleHandler"> <cqtimedatepattern>yyyy-MM-dd
HH:mm:ss</cqtimedatepattern> <cqtimecolumnindex>1</cqtimecolumnindex>
<pumpratio>100</pumpratio> <keeporiginalcqtime>false</keeporiginalcqtime>
</custom>
<pump> <type>COPY</type> <mode>CSV</mode> <schema>my_schema</schema>
<stream>my_stream</stream>
</pump> </chain> </handlers>
```

The handler chain is configured as follows:

- The FileReplayHandler takes the file source_file.txt located in the /tmp directory and allocates 5000 MB cache to store the original data. The cache is then set to be replayed 1000 times.
- The <inputstream/> tag calls the InputStreamHandler to read the source_file.txt text.
- The <text> tag calls the TextHandler to parse the string based on a delimiter parameter. Because this is a comma delimited file, a comma is used as a delimiter.

- The ThrottleHandler then speeds the data rate by 100 times the normal speed. Because the data is replayed, the original timestamp <keeporiginalcqtime> parameter is set to false. However, you can set it to true.
- The Pump handler then takes that input stream and pushes it to the continuous query engine using the COPY command and designating the schema and stream definition defined in the continuous query engine. The pump command is similar to the postgres copy command [-psql -c copy] my_schema.my_stream from '/tmp/source_file.txt' with CSV. The simple copy command pushes data into the system as quickly as possible with the options to replay and/or throttle the source data.

Building a NetFlow Application

NetFlow is a Cisco IOS embedded instrumentation that characterizes network operations. Each packet forwarded within a router or switch is examined for a set of IP packet attributes. These attributes are the IP packet identity or fingerprint. They determine if the packet is unique or similar to other packets. An IP NetFlow is usually based on a set of five and up to seven IP packet attributes:

- IP source address
- IP destination address
- Source port
- Destination port
- Layer 3 protocol type
- Class of Service
- Router or switch interface

All packets with the same source and destination IP address and ports, protocol interface, and class of service are grouped into a flow. The packets and bytes are then tallied. The router or switch sends these tallied flows to a collector to process the data.

The Prime Analytics NetFlow handler receives this information, parses it and feeds it to the CQ engine (TruCQ) for analysis. A handler chain, defined in the file customizations/templates/include-handlers.xml, specifies the input processing path. Here is a sample NetFlow processing chain:

```
<handlers>
  <chain id="netflow">
    <udpnetflow>
      <port>2055</port>
      <rcvbufsize>16000000</rcvbufsize>
    </udpnetflow>
    <netflow queue="true" threads="8">

<fields>UTCtime,routerIP,sequence,input-key,srcaddr-key,srcport-key,output-key,dstaddr-key
,dstport-key,in-packets-key,in-bytes-key</fields>
    </netflow>
    <commitbytes>
      <rows>50000</rows>
    </commitbytes>
    <pumpbytes>
      <type>COPY</type>
      <mode>BYTES</mode>
      <autoflush>false</autoflush>
      <schema>netflow</schema>
      <stream>netflow</stream>
    </pumpbytes>
  </chain>
</handlers>
```

The first handler entry, udpnetflow, indicates the incoming port number and a receive buffer size. NetFlow typically arrives at a high data rate. A large receive buffer prevents packet loss (the UDP protocol does not retransmit dropped packets). The Linux kernel imposes a limit on receive buffer size. The system administrator (root) must increase the limit using a command. For example,

```
sysctl net.core.rmem_max=16000000
```

The next entry, netflow, indicates the fields to be extracted from the received data and sent as columns to the streaming engine. The commitbytes and pumpbytes entries correspond to the <commit> and <pump> entries used with other handlers, but they process binary data instead of strings. The <netflow> handler emits its results in an internal binary form to reduce overhead.

The file customizations/db/ddl.sql specifies an input schema for stream processing. Here is an example that matches the above handler chain:

```
CREATE STREAM netflow (
       tod timestamp with time zone cqtime user drift '1 second',
       routerip text,
       sequence bigint,
       inputif bigint,
       srcaddr text,
       srcport bigint,
       outputif bigint,
       destaddr text,
       destport bigint,
       packets bigint,
       bytes bigint
) checkpoint ;
```

Note the use of bigint for numeric columns. This is required. NetFlow packets include a time of day specified in UTC. The timestamp with time zone adjusts correctly for UTC times.

Note If you change the NetFlow fields that are exported to TruCQ, you must also modify the DDL

Table 2-4 shows the NetFlow field IDs

Table 2-4 NetFlow Field IDs

Field ID (key)	Type (#)	NetFlow V9 Name	Description
UTCtime	n/a	n/a	Seconds since 0000 Coordinated Universal Time (UTC) 1970.
routerIP	n/a	n/a	IP address of origin router.
sequence	n/a	n/a	Packet sequence number (v9) or cumulative flow count (v5).
in-bytes-key	1	IN_BYTES	Incoming counter with length N x 8 bits for number of bytes associated with an IP flow.
in-packets-key	2	IN_PKTS	Incoming counter with length N x 8 bits for the number of packets associated with an IP flow.
flows-key	3	FLOWS	Number of flows that were aggregated.
prot-key	4	PROTOCOL	IP protocol byte.
ip-precedence-key	5	SRC_TOS	Type of Service byte setting when entering incoming interface.
tcp-flags-key	6	TCP_FLAGS	Cumulative of all the TCP flags seen for this flow/
srcport-key	7	L4_SRC_PORT	TCP/UDP source port number, that is, FTP, Telnet, or equivalent.

Table 2-4 NetFlow Field IDs (continued)

Field ID (key)	Type (#)	NetFlow V9 Name	Description
srcaddr-key	8	IPV4_SRC_ADDR	IPv4 source address.
src-mask-key	9	SRC_MASK	The number of contiguous bits in the source address subnet mask, that is, the submask in slash notation.
input-key	10	INPUT_SNMP	Input interface index.
dstport-key	11	L4_DST_PORT	TCP/UDP destination port number, that is, FTP, Telnet, or equivalent
dstaddr-key	12	IPV4_DST_ADDR	IPv4 destination address
dst-mask-key	13	DST_MASK	The number of contiguous bits in the destination address subnet mask, that is, the submask in slash notation
output-key	14	OUTPUT_SNMP	Output interface index
nexthop-key	15	IPV4_NEXT_HOP	IPv4 address of next-hop router
src-as-key	16	SRC_AS	Source BGP autonomous system number
dst-as-key	17	DST_AS	Destination BGP autonomous system number
endtime-key	21	LAST_SWITCHED	System uptime at which the last packet of this flow was switched
starttime-key	22	FIRST_SWITCHED	System uptime at which the first packet of this flow was switched
ipv6-src-addr-key	27	IPV6_SRC_ADDR	IPv6 Source Address
ipv6-dst-addr-key	28	IPV6_DST_ADDR	IPv6 Destination Address
ipv6-flow-label-key	31	IPV6_FLOW_LABEL	IPv6 flow label as per RFC 2460 definition
src-subnet-in-flow-addr-key	44	IPV4_SRC_PREFIX	IPv4 source address prefix (specific for Catalyst architecture)
dst-subnet-in-flow-addr-key	45	IPV4_DST_PREFIX	IPv4 destination address prefix (specific for Catalyst architecture)
sampler-id-key	48	FLOW_SAMPLER_ID	Identifier shown in "show flow-sampler"
sampler-interval-key	50	FLOW_SAMPLER_RANDOM_INTERVAL	Packet interval at which to sample. Use in connection with FLOW_SAMPLER_MODE
classid-key	51	CLASS_ID	
minimum-ttl-key	52	MIN_TTL	Minimum TTL on incoming packets of the flow
maximum-ttl-key	53	MAX_TTL	Maximum TTL on incoming packets of the flow
src-mac-key	56	IN_SRC_MAC	Incoming source MAC address
dst-mac-key	57	OUT_DST_MAC	Outgoing destination MAC address
vlan-id-key	58	SRC_VLAN	Virtual LAN identifier associated with ingress interface
direction-key	61	DIRECTION	Flow direction: 0 - ingress flow, 1 - egress flow
mpls-top-label	70	MPLS_LABEL_1	MPLS label at position 1 in the stack
if-name-in-opt-data-key	82	IF_NAME	Shortened interface name i.e.: "FE1/0"
if-desc-in-opt-data-key	83	IF_DESC	Full interface name, that is, "FastEthernet 1/0"

Table 2-4 NetFlow Field IDs (continued)

Field ID (key)	Type (#)	NetFlow V9 Name	Description
flags-and-sid-key	87	FLAGS_AND_SAMPLER_ID	
application-id-key	95	APPLICATION TAG	8 bits of engine ID, followed by n bits of classification
application-name-key	96	APPLICATION NAME	Name associated with a classification
ASA-flowId	148	flowId	
flow-start-seconds-id-key	150	flowStartSeconds	
flow-end-seconds-id-key	151	flowEndSeconds	
ASA-icmpType	176	transport icmp ipv4 type	
ASA-icmpCode	177	transport icmp ipv4 code	
ASA-icmpTypeIPv6	178	icmpTypeIPv6	
ASA-icmpCodeIPv6	179	icmpCodeIPv6	
dot1qPriority-key	244	dot1qPriority	
ethertype-key	256	ethertype	
ASA-timestampMillis	323	observationTimeMilliseconds	
ASA-ingressACLID	33000	ASA_ingressACLID	
ASA-egressACLID	33001	ASA_egressACLID	
ASA-fwEventDetailed	33002	ASA_fwEventDetailed	
ASA-userID	40000	ASA_userID	
ASA-translatedSrcAddr	40001	ASA_translatedSrcAddr	
ASA-translatedDstAddr	40002	ASA_translatedDstAddr	
ASA-translatedSrcPort	40003	ASA_translatedSrcPort	
ASA-translatedDstPort	40004	ASA_translatedDstPort	
ASA-fwEvent	40005	ASA_fwEvent	
net-encap-key	42010	netEncap	

Building a Syslog Application

Syslog is the standard for logging system events. It separates the applications that generate messages from the system that stores them and the applications that report and analyze them. Syslog protocol is defined by the Internet Engineering Task Force (IETF). The protocol is extendable and the log messages format is customized.

The Prime Analytics syslog handler (SyslogHandler) parses the log messages. The logs have fixed formats including fixed delimiter and timestamp fields. The SyslogHandler receives log messages, parses them, them feeds the messages to the Prime Analytics CQ engine (TruCQ) for analysis.

When creating an application with syslog handler chain, keep the following points in mind:

- SyslogHandler parses common log messages containing lines of strings with fixed delimiters and at least one time stamp field.
- SyslogHandler can import multiple parser rule files and parse the log messages according to the identifier.
- A format definition XML file is needed. The DTD file, syslogparser.dtd, located in customizations/template/handler/syslog/, verifies the parser file format. A SyslogValidator script is located in primeanalytics/Scripts/utils/.
- A parser rule file contains three basic components:
  - <attribute>—includes <delimiter> and <identifier>. <delimiter> is used to split the syslog message into tokens. <identifier> is a user-defined keyword. It can filter the incoming messages, and map the matched ones to the specific parser.
  - <token>—Indicates a raw message substring.
  - <column>—Indicates a stream tuple.
- One token can be extracted by <index>, <regex>, and <kvdelimiter>.
- One column can be empty, one token or combined multiple tokens. It will map the syslog substring into stream tuple. It supports integer/varchar/timestamp/hstore types in PostgreSQL.

Some syslog message examples:

<166>Jul 5 2013 02:47:22 172.20.35.150/Admin %ACE-6-302022: Built TCP connection 0x289111 for vlan28:172.20.35.177/33671 (172.20.35.177/33671) to vlan28:172.20.35.135/443 (172.20.35.135/443)

To build an application based on the SyslogHandler;

Step 1 Define the database schema. This includes writing the DDL and creating the stream. For example:

```
CREATE STREAM syslog(
source inet, -- Source IP
qtime timestamp cqtime user, -- Syslog timestamp
priority varhcar, -- priority defined in Syslog
msgs varchar -- Syslog messages content
) checkpoint;
```

Step 2 Define the include-handler.xml.

The Prime Analytics TruLink component handles input processing. A handler chain, defined in the file customizations/templates/include-handlers.xml, specifies the input processing path. Here is a process syslog example receiving UDP packets:

```
<handlers>
  <chain id="syslog_udp">
    <udpsyslog>
      <port>514</port>
```

```
    </udpsyslog>
    <sylsog>
      <location>definition.xml</location>
    </syslog>
    <commit>
      <rows>50000</rows>
    </commit>
    <pump>
      <type>COPY</type>
      <mode>CSV</mode>
      <schema>syslog</schema>
      <stream>syslog_base</stream>
    </pump>
  </chain>
</handlers>
```

The first handler entry, udpsyslog, indicates the incoming port number and a receive buffer size(optional). Most syslog protocol uses UDP as transport protocol. However, UDP protocol provides no guarantees for message delivery. If the message rate is high, some packets might be lost. A large receive buffer prevents packet loss. The Linux kernel imposes a limit on receive buffer size. The system administrator (root) must increase the limit through a command, for example:

```
sysctl net.core.rmem_max=16000000
```

The next entry, syslog, identifies the definition files that tell how the received data is extracted and sent as columns to the streaming engine. Multiple definition files are supported. These are divided by commas, for example, <location>def1.xml, def2.xml</location>.

SyslogHandler can also read log files from local files, for example:

```
<handlers>
  <chain id="syslog_file">
    <file>
      <location>${syslog.log}</location>
    </file>
    <inputstream/>
    <packetwrapper/>
    <sylsog>
      <location>definition.xml</location>
    </syslog>
    <commit>
      <rows>50000</rows>
    </commit>
    <pump>
      <type>COPY</type>
      <mode>CSV</mode>
      <schema>syslog</schema>
      <stream>syslog_base</stream>
    </pump>
  </chain>
</handlers>
```

The first handler entry, file, reads data streams from the syslog.log, which is defined in local-runtime.properties. The inputstream handler converts input streams into lines. The packetwrapper is a wrapper handler which adapts the syslog line and syslog handler.

Step 3 Define and validate parser rule file.

The parser definition file is required in the Prime Analytics project. It specifies how to parse the message, and how to generate columns. Here is an example:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE parser SYSTEM "customizations/templates/handlers/syslogparser.dtd">
<parser>
```

```
    <attribute>
        <delimiter>" "</delimiter>
        <identifier><![CDATA[*]]></identifier>
    </attribute>
    <token id="token1">
        <index>0</index>
    </token>
    <token id="token2">
        <index>1</index>
    </token>
    <token id="token3">
        <index>2</index>
        <regex>regex</regex>
    </token>
    <token id="token4">
        <index>4</index>
        <kvdelimiter>=</kvdelimiter>
    </token>

    <column>
        <name>column1</name>
        <index>1</index>
        <value>token.token1.1</value>
    </column>
    <column>
        <name>column2</name>
        <index>2</index>
        <value>token.token2.1+" "+token.token3.1</value>
    </column>
    <column>
        <name>column3</name>
        <index>3</index>
        <value>token.token3.1</value>
        <dateformat>MM dd HH:mm:SS.SSS Z</dateformat>
    </column>
    <column>
        <name>column4</name>
        <index>4</index>
        <value>token.token4.n</value>
        <map>key1=value1,key2=value2</map>
    </column>
    <column>
        <name>column5</name>
        <index>5</index>
        <value>token.token4.n</value>
        <hstore>true</hstore>
    </column>
    <column>
        <name>column6</name>
        <index>6</index>
        <value></value>
    </column>
    <column>
        <name>column7</name>
        <index>7</index>
        <value>packet.source.ip</value>
    </column>
    <column>
        <name>column8</name>
        <index>8</index>
        <value>packet.source.port</value>
    </column>
</parser>
```

SyslogHandler Definition File Syntax

- The definition XML format can be validated against the DTD defined in customizations/templates/handlers/syslogparser.dtd:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE parser SYSTEM "customized/templates/handlers/syslogparser.dtd">
<parser>
    <attribute>
        <delimiter>" "</delimiter>
        <identifier><![CDATA[*]]></identifier>
    </attribute>
    <token>
    </token>
    <column>
    </column>
</parser>
```

  All definition file contains three basic elements: attribute, token and column. The attribute contains two sub elements: delimiter and identifier. The delimiter divides the raw syslog messages into tokens. It could be one character, one word or a symbol. The double quotas around the delimiter is optional. The identifier filters the incoming syslog and maps the matched message to the corresponding parser, which is defined in different definition files. Two definition files must be defined for each syslog with unique identifiers. The wildcard means ignore the filters. If multiple definition files are defined, syslog messages are filtered sequentially. The first matched definition file is applied. For example:

```
<identifier><![CDATA[warning]]></identifier>.
```

  In this example, any message containing the word, warning, is filtered and parsed by the current definition file.

- Token is the raw message fragment. It can be referred in column definition. The token node value is an array that includes all matched strings. The mandatory attribute, id, identifies each token:
  - index—The syslog message is divided into an array by the delimiter. The array index starts from 1. The index, 1, is the first substring, and the index, 2, is the second substring. The index "0" is the whole message without division.
  - regex—A regular expression used to extract a string against the token retrieved by the index. It is an optional element. More regular expressions can be found at http://docs.oracle.com/javase/tutorial/essential/regex/.
  - kvdelimiter—Indicates how to divide the string into key/value pair. It is optional.

  In the following example, the delimiter is a whitespace and syslog message is:

```
<166>Jul 5 2013 02:47:22 172.20.35.150/Admin %ACE-6-302022: Built TCP connection
0x289111 for vlan28:172.20.35.177/33671 (172.20.35.177/33671) to
vlan28:172.20.35.135/443 (172.20.35.135/443)

<token id="PRIORITY">
    <index>1</index>       <!-- it refers to "<166>Jul" -->
    <regex>&lt;(\d)&gt;</regex>   <!-- the regex expression extracts from above token
value, the result is 166 -->
</token>

<token id="LEVEL">
    <index>6</index>       <!-- it refers to "%ACE-6-302022:" -->
    <regex>%ACE-(\d+)-\d+</regex>   <!-- the result is "6" -->
</token>
```

```
2013/02/22 09:00:00.019
SdvSmTxSelConf,STB=10.250.82.233,TID=0xef3,SesId=001bd74bbd44/0,RC=OK,MPN=1092,Freq=68
1000000,Mod=16,Tsid=48691,SGID=1309

<token id="detail">
    <index>3</index>               <!-- it refers to
"SdvSmTxSelConf,STB=10.250.82.233,TID=0xef3,SesId=001bd74bbd44/0,RC=OK,MPN=1092,Freq=6
81000000,Mod=16,Tsid=48691,SGID=1309" -->
    <kvdelimiter>=</kvdelimiter>   <!-- the result is
"STB=>10.250.82.233,TID=>0xef3,SesId=>001bd74bbd44/0,RC=>OK,MPN=>1092,Freq=>681000000,
Mod=>16,Tsid=>48691,SGID=>1309" -->
</token>
```

- Column is the corresponding stream column definition. The column is mapped into the stream tuple. It supports varchar, integer, datetime, and hstore. The default is varchar. The column node has the following elements.
  - name—Refers to the column name. It is optional.
  - index —Refers to the column position in the stream. It starts with 1 and is mandatory. If valueIndex is n, it refers to all the values in the token's value array. The column type is array.
  - value—Refers to the column value. It is mandatory. Empty values mean no values are entered for the column. The column value contains token elements, token expressions or packet elements. If it refers to a token element, use token.[tokenid].[valueIndex]. tokenid is defined in the token attribute id. valueIndex refers to the index in the token's value array. If it refers to token expressions, use + to combine multiple token elements. It also could use packet. packet contains two properties: ip and port. These indicate the message source IP and port.
  - dateformat—Refers to the date format used when converting the value into a date. This is mandatory when the column type is date/datetime/timestamp.
  - map—Refers to the key/value pairs that need to be replaced.
  - hstore—Indicates whether the column type is hstore.

  Examples:

```
<166>Jul  5 2013 02:47:22 172.20.35.150/Admin %ACE-6-302022: Built TCP connection
0x289111 for vlan28:172.20.35.177/33671 (172.20.35.177/33671) to
vlan28:172.20.35.135/443 (172.20.35.135/443)

<column>
    <name>datetime</name>
    <index>2</index>
   <value>token.DATETIME_MONTH.1+" "+token.DATETIME_DAY.1+" "+token.DATETIME_YEAR.1+"
"+token.DATETIME_TIME.1</value>   <!-- DATETIME_MONTH is the token id defined in token
element. DATETIME_MONTH.1 means the first value in token DATETIME_MONTH. If
DATETIME_MONTH.n means all the values in that token, and will be converted into
PostgreSQL arraytype -->
    <dateformat>MMM dd yyyy hh:mm:ss</dateformat>   <!-- It refers the format used in
the above value element. If the column is timestamp, user need to specify the date
time format used in syslog message. -->
</column>

<column>
    <name>level</name>
    <index>4</index>
    <value>token.LEVEL.1</index>   <!-- It refers to "6" -->
</column>

2013/02/22 09:00:00.019
SdvSmTxSelConf,STB=10.250.82.233,TID=0xef3,SesId=001bd74bbd44/0,RC=OK,MPN=1092,Freq=68
1000000,Mod=16,Tsid=48691,SGID=1309
```

```
<column>
<name>eventreplace</name>
<index>4</index>
<value>token.detail.n</name> <!-- It refers all the values(key/value pairs) in token
deail -->
<map>SesID=SessionID,RC=ResponseCode</map>
</column> <!-- the reuslt if
{STB=10.250.82.233,TID=0xef3,SessionID=001bd74bbd44/0,ResponseCode=OK,MPN=1092,Freq=68
1000000,Tsid=48691,SGID=1309} -->
```

Step 4 Before you deploy the application, validate the rule files against the sample data. The sample data is a log file.

Run the syslog_validator.sh which you can find in primeanalytics/Scripts/util/. For example:

```
$TRUVISO_HOME/Scripts/util/syslog_validator.sh sample.log definition.xml syslogparser.dtd
```

- Argument 1—Is the input sample syslog messages.
- Argument 2—Is the parser file path.
- Argument 3—Is optional, the default DTD file can be read specified in the definition.xml.

Building Multistream Handlers

A typical Prime Analytics application expects data arriving at the TruCQ stream-processing engine from a specific input source to have a consistent structure. A one-to-one correspondence usually exists between a data source and the application raw stream schema. However, some data sources produce records with varying structures. In such cases, you might want to route the input records from a single data source to multiple raw streams. The multistream handler gives you the ability to divide a single data source into multiple streams.

The division is based on a specific partitioning key in the input stream. This partitioning key is specified in a handler chain that includes the multistream handler as a list of pairs (key_value, raw_stream_name). The list provides the record-to-stream mapping; it identifies the specific stream any given record must be directed to based on the value of the partitioning key attribute.

The multistream handler has the following requirements:

- The multistream handler must be the last handler in a chain—It must always be placed at the end of a handler chain. No further record preprocessing can occur after determining a record's destination raw stream and pumping the record to the stream.
- Mapping is based on key values—The multistream handler routes records to streams based on record key-attribute values. User-defined data routing functions and use of regular expressions as pump keys are not supported.

Multistream Handler Example

The multistream handler in the example below routes input records having my_key values H, M, and B to the raw streams. raw_m, raw_h, and raw_b. The handler sends the records having any other my_key values to the raw stream raw_default.

The multistream handler example configuration file is shown below:

```
Include-handlers.xml
<chain id="EXAMPLE" autostart="true">
    <file>
```

```
        <location>${master.file}</location>
    </file>
     <inputstream />
     <text>
        <delimiter>\t</delimiter>
     </text>
     <multistream>
        <pumpkeyindex>1</Pumpkeyindex>  <!-- Specifies the location of the key in an input
record. In this example, we choose as key the field my_key which is at location 1 in a
record that is input to the multistream handler-->
         <pumps> <!-- A list of key-pump pairs such that
                each pump pushes data to its specified
                raw stream -->
             <pump key="M">
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>multistream_example</schema>
                 <stream>raw_m</stream>
            </pump>
            <pump key="H">
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>multistream_example</schema>
                 <stream>raw_h</stream>
             </pump>
             <pump key="B">
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>multistream_example</schema>
                 <stream>raw_b</stream>
             </pump>
            <pump key="*">   <!-- Optional. A pump key entry "*" means that a record with
the key-attribute value outside of any of specified values (M, H, and B, in this example)
should go to the stream specified in this entry (raw_default, in this example).-->
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>multistream_example</schema>
                 <stream>raw_default</stream>
             </pump>
        </pumps>
     </multistream>
</chain>
```

The multistream handler schema is shown below:

```
my_ddl.sql
    create schema multistream_example;
    set search_path to multistream_example;
    - - multiple raw streams
create stream raw_m
(
     my_key Text,
     a int
);
create stream raw_h
(
    my_key Text,
    a   int,
    ts  timestamp cqtime user
);
create stream raw_b
(
    my_key Text,
    msg Text
```

```
);
create stream raw_default
(
    my_key Text,
    msg Text
);
    - - a few derived streams

create stream h_derived (total, ts cqtime)
checkpoint without partials as
select sum(a), cq_close(*) from raw_h
<slices '1 minute'>;

create stream b_derived (message_count)
checkpoint without partials as
select count(key) from raw_b
<slices '1 minute'>;
```

Building an XMPP Connector

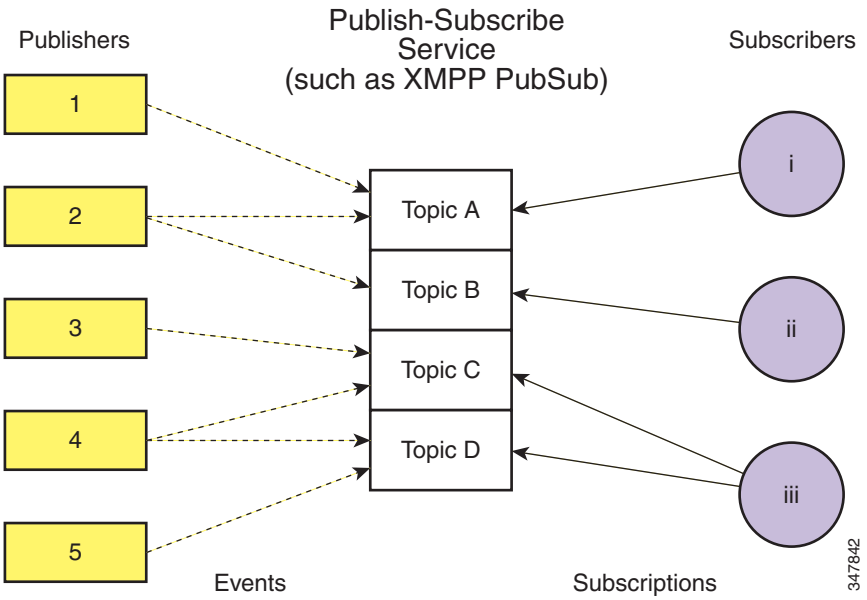
XMPP is the eXtensible Messaging and Presence Protocol, a set of open technologies for instant messaging, presence, multi-party chat, voice and video calls, collaboration, lightweight middleware, content syndication, and generalized routing of XML data.

XMPP was originally developed in the Jabber open-source community to provide an open, secure, spam-free, decentralized alternative to the closed instant messaging services at that time.

XMPP PubSub is a protocol extension for generic publish-subscribe functionality, specified in XEP-0060. The protocol enables XMPP entities to create nodes (topics) at a pubsub service and publish information at those nodes; an event notification (with or without payload) is then broadcast to all entities that have subscribed to the node. Pubsub therefore adheres to the classic observer design pattern and can serve as the foundation for a wide variety of applications, including news feeds, content syndication, rich presence, geo-location, workflow systems, network management systems, and any other application that requires event notifications.

Figure 2-1 shows the XMPP publish-subscribe service flow.

Figure 2-1 XMPP Publish-Subscribe Service Flow



Prime Analytics only supports the XMPP pubsub mechanism and is tested with two products: Cisco Conductor for Videoscape and Openfire, an instant messaging and groupchat server that uses XMPP licensed under the Apache License 2.0.

The Prime Analytics XMPP handler functions as the subscriber. It receives the XMPP messages and parses the payload using the XML handler.

To connect to the pubsub server, you must configure the communications parameters listed in Table 2-5.

Table 2-5 XMPP Communications Parameters

Parameter	Description
xmpphostname	The Cisco Conductor server/pubsub server address.
port	The standard port for clients to connect to the server.
subscribeuser	User with subscribe access.
password	Password.
isDebug	Enables the Smack debug window.
pubsubId	Pubsub service name.
pubsubNode	A virtual location to which information can be published and from which event notifications and/or payloads can be received. (In other pubsub systems, this may be labeled a “topic”.)

XSLT-Based XML Handler

By default, you can use the XSLT to parse the payload, so you need to specify the XSLT path for XML parsing. For example:

```
<handlers>
   <chain id="xmpp">
      <xmpp>
        <xmpphostname>10.74.125.169</xmpphostname>
        <port>5222</port>
```

```
        <subscribeuser>pub@crdc-c210-169/conductor</subscribeuser>
        <password>pub</password>
        <isDebug>true</isDebug>
        <pubsubId>pubsub.crdc-c210-169</pubsubId>
        <pubsubNode>cisco.pubsub</pubsubNode>
      </xmpp>
      <xml>
        <xslt>/var/opt/primea/projects/acal/ACAL_Setup.xslt</xslt>
      </xml>
      <text>
         <delimiter>,</delimiter>
      </text>
      <commit>
         <rows>200</rows>
      </commit>
      <pump>
         <type>COPY</type>
         <mode>CSV</mode>
         <schema>acal</schema>
         <stream>setup</stream>
      </pump>
   </chain>
</handlers>
```

Multiple Pubsub Nodes Support

The XMPP handler uses multiple chains to support multiple nodes and topics. Each message type is defined in one chain, and the XSLT contains the node and topic name. The following examples shows two nodes, truviso and primea:

```
<handlers>
   <chain id="xmpp">
      <custom>
        <xmpphostname>10.75.162.230</xmpphostname>
        <port>5222</port>
        <subscribeuser>utruviso@client.com</subscribeuser>
        <password>cisco123</password>
        <isDebug>true</isDebug>
        <pubsubId>ps.com</pubsubId>
        <pubsubNode>truviso</pubsubNode>
      </custom>
      <custom>
        <xslt>/var/opt/primea/projects/xmpp/transform_truviso.xslt</xslt>
      </custom>
      <text>
         <delimiter>,</delimiter>
      </text>
      <pump>
         <type>COPY</type>
         <mode>CSV</mode>
         <schema>xmpp</schema>
         <stream>tvshow</stream>
      </pump>
   </chain>
   <chain id="xmpp_primea">
      <custom>
        <xmpphostname>10.75.162.230</xmpphostname>
        <port>5222</port>
        <subscribeuser>uprimea@client.com</subscribeuser>
        <password>cisco123</password>
        <isDebug>true</isDebug>
        <pubsubId>ps.com</pubsubId>
```

```
        <pubsubNode>primea</pubsubNode>
      </custom>
      <custom>
        <xslt>/var/opt/primea/projects/xmpp/transform_primea.xslt</xslt>
      </custom>
      <text>
         <delimiter>,</delimiter>
      </text>
      <pump>
         <type>COPY</type>
         <mode>CSV</mode>
         <schema>xmpp</schema>
         <stream>tvshow</stream>
      </pump>
   </chain>
</handlers>
```

One Pubsub Node With Multiple Topics

One pubsub node could send out multiple formats of messages or payload, such as the setup and tear down messages are sent to the same pubsub node. Take the ACAL setup and tear down messages as the example. Two part should be updated, one is XSLT and another is the db pump part.

- Update the XSLT to diff the message parsing. Different messages have different fields to parse:

```
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
                <xsl:template match="ACAL/report">
            <xsl:choose>
            <xsl:when test="@type='setup'">
            <xsl:value-of select="@type" />,<xsl:value-of select="@id"
/>,<xsl:value-of select="@seq" />,<xsl:value-of select="@time" />,"<xsl:value-of
select="label" />","<xsl:
value-of select="assetName" />",<xsl:value-of select="assetSrc" />,<xsl:value-of
select="endpointID" />,<xsl:value-of select="platformClass" />,<xsl:value-of
select="platformMak
e" />,<xsl:value-of select="platformModel" />,<xsl:value-of
select="platformApplication" />,<xsl:value-of select="platformAppVer" />,</xsl:when>
            <xsl:when test="@type='teardown'">
            <xsl:value-of select="@type" />,<xsl:value-of select="@id"
/>,<xsl:value-of select="@seq" />,<xsl:value-of select="@time" />,<xsl:value-of
select="reason" />,</xsl:w
hen>
            </xsl:choose>
                </xsl:template>
</xsl:stylesheet>
```

- Update configuration to include the Multiple Steam handler, the two kinds of fields are pumped into different streams. For more information, see Building Multistream Handlers, page 2-17.

```
<handlers>
   <chain id="xmpp">
      <xmpp>
        <xmpphostname>172.20.127.147</xmpphostname>
        <port>5222</port>
        <subscribeuser>utruviso@client.com</subscribeuser>
        <password>cisco123</password>
        <isDebug>true</isDebug>
        <pubsubId>ps.com</pubsubId>
        <pubsubNode>truviso</pubsubNode>
      </xmpp>
      <xml>
        <xslt>ACAL_Setup_C.xslt</xslt>
      </xml>
```

```
        <text>
           <delimiter>,</delimiter>
        </text>
        <commit>
           <rows>2</rows>
        </commit>
        <multistream>
           <pumpkeyindex>1</pumpkeyindex>
           <pumps>
              <pump key="setup">
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>acal</schema>
                 <stream>setup</stream>
              </pump>
              <pump key="teardown">
                 <type>COPY</type>
                 <mode>CSV</mode>
                 <schema>acal</schema>
                 <stream>teardown</stream>
              </pump>
           </pumps>
        </multistream>
     </chain>
  </handlers>
```

Troubleshooting Handlers and Connectors

If no data is received by the connector or handler, check network connectivity:

1. Check the physical connection (most network interfaces have a green light indicating the link is active).
2. Check the network configuration including IP address, subnet mask, and default route.
3. Check the firewall (iptables).
4. Test connectivity using ping.
5. Use tcpdump to check for packets received.
6. Verify the port number in the application settings.
7. For HA, check that multicast is allowed. (For information, see http://www-01.ibm.com/support/docview.wss?uid=isg3T1012468.)

If you receive an “ERROR: extra data after last expected column” while using the XSLT-based XML handler, check whether a comma or special character is the same delimiter defined in text part. It parses original messages into several fields. This causes the database exceptions because there are more fields than the db schema defined, so the delimiter within the message is ignored.

```
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:df="http://jabber.org/protocol/pubsub">
                <xsl:template match="/">
                                <xsl:for-each select="df:ACAL/df:report">
            <xsl:value-of select="@type" />,<xsl:value-of select="@id" />,<xsl:value-of
select="@seq" />,<xsl:value-of select="@time" />,"<xsl:value-of select="df:label" />","<x
sl:value-of select="df:assetName" />",<xsl:value-of select="df:assetSrc" />,<xsl:value-of
select="df:endpointID" />,<xsl:value-of select="df:platformClass" />,<xsl:value-of sele
```

```
ct="df:platformMake" />,<xsl:value-of select="df:platformModel" />,<xsl:value-of
select="df:platformApplication" />,<xsl:value-of select="df:platformAppVer"
/>,</xsl:for-each>
                </xsl:template>
```

Creating Continuous Queries

Continuous queries are persistent inquiries that operate over streams and tables. Continuous queries are created through XML definition files.

To validate the continuous query XML configuration file:

Step 1 In the data.xml file verify the following is present in the <!DOCTYPE.> section:

```
<!ENTITY connectors SYSTEM "include-queries.xml">
```

Step 2 In the <config> section, but within another tag, verify the following is present:

```
&queries;
```

This allows you to configure all continuous queries in the include-queries.xml file rather than within the data.xml file itself.

Step 3 Verify that the include-queries.xml file is in the same directory as the data.xml file.

Step 4 Edit the include-queries.xml file to include the parameters and definitions required by the continuous queries.

Creating Static Queries

Static queries operate over data dimension tables and archived continuous query data tables. Static queries are the same as traditional relational queries. Like continuous queries, static queries are created through XML definition files.

To validate the XML configuration file for static queries:

Step 1 In the data.xml file verify the following is present in the <!DOCTYPE.> section:

```
<!ENTITY connectors SYSTEM "include-staticqueries.xml">
```

Step 2 In the <config> section, but within another tag, verify the following is present:

```
&staticqueries;
```

This allows you to configure all continuous queries in the include-staticqueries.xml file rather than within the data.xml file itself.

Step 3 Verify that the include-staticqueries.xml file is in the same directory as the data.xml file.

Step 4 Edit the include-staticqueries.xml file to include the parameters and definitions required by the static queries.

Compiling the Prime Analytics Project

After you have configured the Prime Analytics connectors, handlers, and the continous or static queries, you must compile and then run your project.

To compile your project, enter:

```
$ ./build-local.sh
```

If the build compiles with no errors, the following directories are created in the project's root directory:

- build
- dist
- war

These directories contain the runtime distribution of the your Prime Analytics project.

Starting and Monitoring the Project

After your project is compiled, verify the TruCQ engine is started and any associated DDL has been executed. Then enter the following commands to start and stop your project:

To start the project

```
./start.sh
```

To stop the project:

./stop.sh

To monitor your application activity, review the log files in apache-tomcat/logs/. The truviso.log is the primary log file. However, some startup errors might only appear in the catalina.out file. Some problems that can prevent application startup might be better analyzed by looking at errors in the database log files:

```
tail $CQLOG
```

Where $CQLOG is the directory containing the CQ engine database log.

To view your data using the Prime Analytics BI platform, open your browser and navigate to http://servername:8080/*MyProject* Log in with the default credentials: admin/admin. You can leave the locale at its default.

For information about creating dashboards and reports from the BI platform, see the following topics:

- Navigating the Prime Analytics User Console
- Creating Dashboards
- Creating Reports
- Creating an Analysis

Navigating the Prime Analytics User Console

After you connect the data source to Prime Analytics, you start the Prime Analytics user console and begin reviewing the data. The following topics tell you how to log into the user console and how to navigate to different data review and analysis options:

Logging Into the User Console

Follow these steps to log into the Prime Analytics user console:

Step 1 Start a Web browser session.

Step 2 In the browser URL field, enter the Prime Analytics server IP address or hostname and connection port 8080: *yourserver*:8080.

Note Port 8080 is the default; this can be changed during installation.

Step 3 In the login window, enter your username and password, then click **Log in**.

You are now logged into the Prime Analytics user console. Proceed to one of the following topics:

- Navigating the User Console, page 3-2

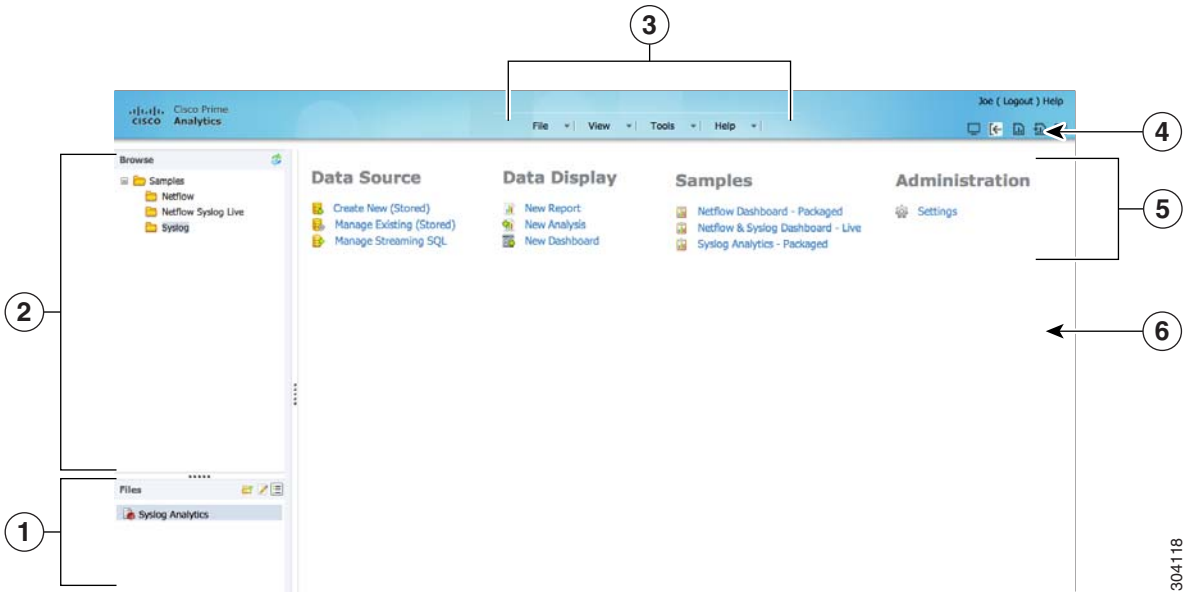
Navigating the User Console

The Prime Analytics user console is the starting point for displaying your project data in many different ways. The user console contains the following elements:

- Browse—Displays a directory containing your project files. you can create new directories to store your data files, including reports, dashboards, and analytics files. You can add, delete, or move directories as you would any standard directory. The NetFlow and Syslog directories provided in the default installation contain sample files that you can use to build your own NetFlow and syslog data files.
- Files—Clicking a directory displays the files it contains here. Selecting a file display icons that you can use to perform actions on the file: Open, Edit, Open in New Window, Run in Background, Delete, Share, Schedule, and Properties.
- Menus—Contain items that invoke all user console actions:
  - File—Provides all file management options: creating new files, saving files, displaying file properties, scheduling, sharing, and other file management actions. Every data view you create, whether a new report, a new dashboard, or a new analysis, is considered a file, so all standard file actions, save, save as, edit, can be performed on them.
  - View—Provide options for viewing your projects and project files. Browser displays the default web browser view; Workspace displays the status of reports you submitted to run in the background. Other options are defaults that canot be changed.
  - Tools—Provides options to refresh of all BI elements, from the repository cache and systems settings to global variables and reporting data cache. The CDA cache manager is also accessed from this menu.
  - Help—Provides links to Prime Analytics help.
- Main toolbar—Provides tools that invoke common actions:
  - Workspace—Opens the Workspace window. The Workspace screen will report what you have submitted to run in the background on the server. Within this screen, you can cancel reports that have not run yet, and you can view or delete ones that have. This action can be achieved by going to View -> Workspace. You can also toggle between the Workspace view and the Browser view within the View menu.
  - Toggle Browser—This button will allow you to show or hide the Browse Pane.
  - New Report—Opens the Report tab where you can create a new report.
  - New Analytics—Opens the Analytics tab where you can create a new analysis.
  - New Dashboard—Opens the Dashboard tab where you can create a new dashboard.
- Browse Pane - The browse pane on the left allows you to navigate through your existing solution folders. Solutions and content files such as dashboards and reports reside inside the folder. You can create and delete solution folders by right-clicking inside the navigation pane. You can also examine information associated with solution folder contents by right-clicking and selecting Properties.
- Work area—Displays your report, dashboard, or analysis project data. If no files are open, the following menus are displayed:
  - Data Source—Provides options to create new and manage existing data sources. For information, see Chapter 4, “Setting Up the Data Sources.”
  - Data Display—Provides options that allow you to create and manage the display of Prime Analytics data. For information, see Chapter 6, “Creating Reports,” Chapter 7, “Creating an Analysis,”and Chapter 5, “Creating Dashboards.”

- Samples—Allows you to display the sample applications provided with Prime Analytics. For information, see Appendix B, "Sample Applications."
- Administration—Displays a Settings option that allows you to modify Prime Analytics settings. For information, see Chapter 8, "Managing Prime Analytics."

Figure 3-1 Prime Analytics User Console



1	Files contained in the selected folder	**4**	User Console toolbar
2	Folders	**5**	Menus
3	User Console menus	**6**	Work Area

The Prime Analytics user console is a starting point for creating and managing your data files. For more detailed information, see one or more of the following:

- Chapter 5, "Creating Dashboards."
- Chapter 6, "Creating Reports."
- Chapter 7, "Creating an Analysis."

Setting Up the Data Sources

After you install Prime Analytics and set up your project, you next must define and connect to your data sources. Setting up data source connectivity is covered in the following topics

Adding the JDBC Driver

Before you can connect to a data source in any Prime Analytics server or client tool, you must install the appropriate database driver. Your administrator should provide you with the proper driver JAR. If not, download the JDBC driver JAR file from your database vendor or developer website. After you have the JAR, follow the instructions below to copy the JAR to the driver directories for all of the components that you want to connect to this data source.

Note Microsoft SQL Server users frequently use an alternative, non-vendor-supported driver called JTDS. If you are adding an MSSQL data source, verify that you are installing the correct driver.

Backing Up Old drivers

Verify that no other versions of the same vendor JDBC driver are installed in the same directory. If so, back up the driver, then remove it to avoid confusion and potential class loading problems.

Installing the JDBC Drivers

To install the JDBC driver:

Step 1 Copy the driver JAR file to the following directories, depending on which servers and client tools you are using:

- BI Server:

  $PA_HOME/primeanalytics/biplatform/biserver-ce/tomcat/lib/

- Administration Console:

  $PA_HOME/primeanalytics/biplatform/administration-console/jdbc/

- Schema Workbench:

  $PA_HOME/primeanalytics/biplatform/schema-workbench/drivers/

Step 2 Once the driver JAR is installed, restart the server or client tool that you added it to.

Creating Historical Data Sources

Setting up connectivity to historical data sources is accomplished using the Data Source Wizard. To set up data source connectivity:

- Adding a CSV File Data Source, page 4-2
- Creating an SQL Query Data Source, page 4-4

Adding a CSV File Data Source

Follow the instructions below to add to a data source based on a CSV file.

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **New > Data Source**.

Step 3 In the Data Source Wizard, Data Source Name field, enter the new data source name.

Step 4 Under Source Type, choose **CSV File**.

Step 5 Click **Import**.

Step 6 In the Import File dialog, click **Choose File**.

Step 7 In the File Upload dialog box, navigate to your CSV file, then click **Open**.

Step 8 In the Import File dialog, click **Import**.

The Import File dialog box closes and the CSV file is uploaded to the server. (If you do not want to continue, you can click **Cancel** and select different file.)

A preview of your file contents appears in the lower portion of the wizard page. If items are not aligning correctly, it is because you must specify your delimiter and enclosure types. Go on to the next step.

Step 9 Specify the following CSV parameters:

- Delimiter—Specifies how your file contents is delimited: Comma (default), Tab, Semicolon, Space, Other (allows you to enter a different delimiter item.
- Encoding—Specifies the CSV encoding. The default is ISO-8859-1.
- Enclosure—Specifies the enclosure: Double Quote, Single Quote, or None.
- First row is Header—If checked (default), the first row of your CSV file is a heading for columns in the file.

File Preview displays the first lines of your CSV file based on your delimiter, enclosure, and header selections. Column alignment indicates the delimiter and enclosure are set correctly.

Step 10 Click **Next**.

The wizard Staging Setting panel allows you edit the final display.

Step 11 As needed, edit the final file display:

- All columns are enabled by default. If you do not want to include all of them, uncheck the ones you want to exclude. You can use Deselect All and Select All to disable and enable all columns.
- The Name and Type columns display the provided column name and type. You can edit these as necessary.
- The Source Format field displays the source format for dates and numeric values. Formats are automatically detected based on the your data source contents. The drop-down list displays alternate format options that you can select. Drop-down lists are not enabled for certain data types such as the String data type.
- Length indicates the maximum number of characters allowed in a field.
- Precision is the number of digits after a decimal point.

Note Boolean values are rendered as true or false.

- Click **Show File Contents** to see a sample of the data in your source file. Click **OK** to return to staging and make changes if necessary.

Step 12 After you complete your configuration and formatting review, click **Finish**.

A success message indicates the number of rows being loaded into the database and that a default metadata model was created.

Step 13 Click one of the following:

- Keep default model—If you want use the default model or,
- Customize model now—To customize the model.

Your CSV file data is extracted and staged in a database table. The new data source is now available for use in reports, dashboards, and analyses.

Creating Database Connections

Prime Analytics SQL and database queries require you to first create a connection to the host containing the database, including port, host name, username, and password.

To create a database connection:

Step 1 Verify that the JDBC driver is added. For information, see Adding the JDBC Driver, page 4-1.

Step 2 Log into the Prime Analytics user console as an administrator user. For procedures, see Logging Into the User Console, page 3-1.

Step 3 From the File menu, choose **New > Data Source**.

The Data Source Wizard appears.

Step 4 In the Data Source Name field, enter the new data source name.

Note The data source name is displayed in analytics, reports, and dashboards.

Step 5 Under Source Type, choose **SQL Query** or **Database Table(s)**.

Step 6 From the Connection toolbar, choose **Add Connection**.

Step 7 In the Database Connection dialog, enter the following information:

- Connection Name—Enter a name for your connection.
- Database Type—Two database types are available:
  - TruCQ—Prime Analytics TruCQ query database if archiving is enabled.
  - Generic—A generic database.
- Access (not editable)—Access is enabled using JDBC drivers.
- Host Name—Enter the name of the host where your database is located.
- Database Name—Enter the database name.
- Port Number—Enter the port number used to access the host.
- User Name—Enter the username
- Password—Enter the password.

Step 8 After you enter the connection information, click **Test** to test it.

If a connection is established, a Test Connection dialog indicates the connection to [*database name*] is OK. It also lists the hostname, port, and database name. If the connection was not successful, review your connection details and correct any errors.

Step 9 Click **OK**.

Step 10 Continue with one of the next procedures:

- Creating an SQL Query Data Source, page 4-4
- Creating a Database Table Data Source, page 4-5

Creating an SQL Query Data Source

The following procedure tells you how to create an SQL query from an SQL database. To create it, you must be connected to the database using JDBC. You will enter an SQL statement that defines the SQL data source scope.

You can customize the data columns that are presented to users who build SQL queries. For example, you can define column names and select options that define how data is aggregated (sum, min., max., etc.). After you create the SQL query, it is available to users use in reports, dashboards, and analyses.

Note To complete this procedure, the SQL database must be up and running.

Step 1 Verify that you are connected to the database where you want to create a query. To create a connection, see Creating Database Connections, page 4-3.

Step 2 Log into the Prime Analytics user console as an administrator user. For procedures, see Logging Into the User Console, page 3-1.

Step 3 From the File menu, choose **New > Data Source**.

The Data Source Wizard appears.

Step 4 In the Data Source Name field, enter the new data source name.

Note The data source name is displayed in analytics, reports, and dashboards.

Step 5 Under Source Type, choose **SQL Query**.

Step 6 Under Connection, choose the database connection for the database where you want to run your query.

Step 7 Under SQL Query, enter your SQL query. For example, to select certain columns:

```
SELECT Device, Interface, IPaddress FROM NetworkDevices
```

Or, to select all data:

```
SELECT * FROM NetworkDevices
```

Step 8 Click Data Preview to verify that your query returns data.

Step 9 Click **Finish**.

The query is added.

Creating a Database Table Data Source

You can create a reporting and/or analysis data source against one or more relational database tables. This data source type is useful for a broad range of use cases including reporting directly against an operational data store (ODS) or creating a connection for an online analytical processing (OLAP) analysis against a star schema. When you can create the database table(s) data source, two options are provided:

- Reporting Only—If you access data in a relational database that is operational or transactional, or which does not contain a star schema, use the Reporting Only option. It creates a data source that can be used in reports and dashboards. A metadata model is automatically generated. Users can access the data to create reports and build dashboards.
- Reporting and Analysis—If you access a relational database containing tables arranged in a star schema, that is, tables separated into dimensions related to a single fact-table, such as a data warehouse, use the reporting and analysis option. You can also create a reporting and analysis data source using a single table where the table acts as the single fact table and also contains dimensional information. This option creates a data source that can be used in analytics, reports, and dashboards. A Mondrian schema and metadata model are generated

The choice you make depends on the schema structure of the database tables you are accessing and the user console tools you are using. The following procedures tell you how to create a table data source for reporting only and for reporting and analysis:

- Creating a Database Table Data Source for Reporting Only, page 4-5
- Creating a Database Table Data Source for Reporting and Analysis, page 4-7

Creating a Database Table Data Source for Reporting Only

Follow the instructions below to create a database table(s) data source for reporting.

Note To complete this procedure, the database must be up and running.

Step 1 Verify that you are connected to the database table(s) where you want to create a query. To create a connection, see Creating Database Connections, page 4-3.

Step 2 Log into the Prime Analytics user console as an administrator user. For procedures, see Logging Into the User Console, page 3-1.

Step 3 From the File menu, choose **New > Data Source**.

The Data Source Wizard appears.

Step 4 In the Data Source Name field, enter the new data source name.

Note The data source name is displayed in reports and dashboards.

Step 5 Under Source Type, choose **Database Table(s)**.

Step 6 Under Connection, choose a database connection.

Step 7 At the bottom of the page, click, **Reporting only**, then click **Next**.

A list of available tables retrieved from your database is displayed.

Step 8 Under Available Tables, choose a table and click the right green arrow to move it under Selected Tables.

To choose multiple tables press **CTRL**. To remove a table, chose it and click the left green arrow.

Step 9 When finished, click **Next**.

Note While you can create a data source from a single table, the more common use is selecting and joining multiple tables.

Step 10 In the Define Joins panel, define the table relationships:

a. In the Left Table list, choose the table that you want to join. When you choose a table, key fields appear under Key Field.

b. In the Right Table list, choose the table that you want join to the table on the left.

c. Click **Create Join** to define the inner join between each table.

Note The only join condition allowed in the Data Source Wizard is an inner join; references to left table and right table are not associated with other types of join conditions.

Step 11 Click **Finish**.

A default metadata model is generated in the background for use in reports and dashboards. You can choose to keep the default model or customize the it.

Creating a Database Table Data Source for Reporting and Analysis

Follow the instructions below to create a database table(s) data source for reporting and analysis.

Note To complete this procedure, the database must be up and running.

Step 1 Verify that you are connected to the database table(s) where you want to create a query. To create a connection, see Creating Database Connections, page 4-3.

Step 2 Log into the Prime Analytics user console as an administrator user. For procedures, see Logging Into the User Console, page 3-1.

Step 3 From the File menu, choose **New > Data Source**.

The Data Source Wizard appears.

Step 4 In the Data Source Name field, enter the new data source name.

Note The data source name is displayed in analytics, reports, and dashboards.

Step 5 Under Source Type, choose **Database Table(s)**.

Step 6 Under Connection, choose a database connection.

Step 7 At the bottom of the page, click, **Reporting and Analysis**, then click **Next**.

Step 8 Under Schema, choose the schema you want to apply to the table.

A list of available tables retrieved from your database is displayed.

Step 9 Under Available Tables, choose a table and click the right green arrow to move it under Selected Tables.

To choose multiple tables press **CTRL**. To remove a table, chose it and click the left green arrow.

Step 10 In the Fact Table, choose the tables you want to serve as the fact table.

Step 11 When finished, click **Next**.

Note While you can create a data source from a single table, the more common use is selecting and joining multiple tables.

Step 12 In the Define Joins panel, define the table relationships:

a. In the Left Table list, choose the table that you want to join. When you choose a table, key fields appear under Key Field.

b. In the Right Table list, choose the table that you want join to the table on the left.

c. Click **Create Join** to define the inner join between each table.

Note The only join condition allowed in the Data Source Wizard is an inner join; references to left table and right table are not associated with other types of join conditions.

Step 13 Click **Finish**.

A default metadata model is generated in the background for use in reports and dashboards. You can choose to keep the default model or customize it.

Setting Up Continuous Query Data Sources

Complete the following procedure to set up data sources for real-time (continuous query) data:

Note To complete this procedure, the database must be up and running.

Note This procedure only applies to TruCQ stream connections. To connect to an archived database table, see Creating Database Connections, page 4-3.

Step 1 Verify that the TruCQ engine is running by entering the following command as the root user:

```
service trucq status
```

Note If you switch to the root user, make sure you use the su - root command to ensure that the appropriate environment variables are loaded.

If the TruCQ engine is not running, you will need to start it. See Starting and Stopping Prime Analytics Servers, page 8-30, for information.

Step 2 Stop the BI platform:

```
service biplatform stop
```

Step 3 Navigate to the following directory:

```
$PA_HOME/biplatform/biserver-ce/tomcat/webapps/pentaho/WEB-INF/classes
```

Step 4 Using an XML editor, open the data.xml file in this directory.

Under <trucq></trucq> tag, you will find two pre-existing connections.

Step 5 Copy the existing trucq connections and paste at the end of the xml before the </engines> tag, for example:

```
<trucq id="netflow">
<engine id="netflow" maxSharedConnections="5">
                <url>jdbc:truviso://localhost:5432/cqdb</url>
                <username>primea </username>
                <password></password>
                <driver>com.truviso.Driver</driver>
            </engine>

        </trucq>
</engines>
```

Step 6 Modify the details according to your environment:

- Trucq Id—Provide a unique engine name. This name must be unique in the file.

- Engine id—Provide a unique engine name. This is the name that will appear in the Add SQL dialog box Select a Connection list when you create the continuous query. The name must be unique in the file.
- URL—Enter the JDBC URL in the format:

  ```
  jdbc:truviso://<trucq engine address>:<trucq port>/<trucq database>
  ```

- Username—Enter the database username.
- Password—Enter the database password.
- Driver—Enter the TruCQ driver name in the format: driver=com.truviso.Driver

Step 7 Save the changes.

Step 8 As the root user, start the biplatform:

```
service biplatform start
```

Step 9 Verify the new entry:

a. Log into the BI platform as an administrator user. (See Logging Into the User Console, page 3-1.)

b. Click Manage Streaming SQL.

c. Click Add SQL

d. Verify the new engine ID is available the Select a Connection list.

If the new engine ID appears, continue with the next step. If not, for example, if errors appear when you click Manage Streaming SQL or your connection does not appear:

a. Check the data.xml to verify the details are correct. Try connecting to the database using the psql commandline tool.

b. Check the logs under $PA_HOME/biplatform/biserver-ce/tomcat/logs/catalina.out or pentaho.log or truviso.log.

c. Repeat Steps 1 through 9.

Step 10 Log into the Prime Analytics user console as an administrator user.

Step 11 If any files are open, close them to display the Prime Analytics home page.

Step 12 Under Data Source, click **Manage Streaming SQL**.

Note If you are not connected to the real time data, an error is displayed when you click Manage Streaming SQL.

Step 13 On the Real Time tab, click **Add SQL**.

Step 14 In the Add SQL dialog box, complete the following information:

- Select a Connection—Choose a connection from the available list.
- Provide CDA File Name—Enter the Community Database Access file name.
- Provide a Location for CDA—Enter the file location, or click the browse icon and navigate to the file location.
- Provide a SQL—Enter the SQL you want to use on the data, for example:

  ```
  SELECT cq_close(*), sum(packets) as packets from                    \
  jellybeans.netflow_v5 <slices '1 second'> where routerip=?
  ```

Note Verify the SQL command using the psql command line tool before entering it here.

- Description—Enter a description for your query.
- Query Type—Choose the basis for the query type:
  - Time Based
  - Row Based

  Each continuous data stream has a CQTIME attribute. This attribute can be of the following types: smallint, int, bigint, timestamp without time zone, and timestamp with time zone. If CQTIME is defined as timestamp without time zone or time stamp with time zone, it is considered time-based.
- Parameter Present—If you want to add parameters, choose Yes. Otherwise, choose No.
- Parameters—If you want to add parameters, enter the following:
  - Parameter Column Name
  - Parameter Column Data Type
  - Parameter Test Value

  Note The test value is only used to check the query validity. The actual value is entered when you execute the dashboard.

Step 15 When finished, click **Create**.

The query is validated. If the query has errors, messages are displayed. If this occurs, correct the error and save it again.

Creating Dashboards

The following topics tell you how to create and manage Prime Analytics dashboards:

Overview to Prime Analytics Dashboards

Dashboards allow you to display your data in many different ways so that you can interpret data quickly and effectively. Prime Analytics dashboards are built using open source components and applications, including the Pentaho CTools framework. CTools applications and tool sets include

- Community Dashboards Framework (CDF)
- Community Chart Components (CCC)
- Community Data Access (CDA)
- Community Dashboard Editor (CDE)

Pentaho CTools information and documentation is available from the Pentaho Community website: *http://community.pentaho.com/*.

Creating a dashboard in Prime Analytics is accomplished using the following processes:

1. Lay out a grid for your dashboard.
2. Identify the data sources to be visualized in the dashboard.
3. Create the dashboard visual components, connect them to the data sources, and assign them to the layout grid.
4. Improve and modify the dashboard; add flexibility by parameterizing components and letting components interact with one another.

The steps can be completed in any order. In fact, beginning with a basic dashboard and then adding additional components and interactivity works best as you gain experience.

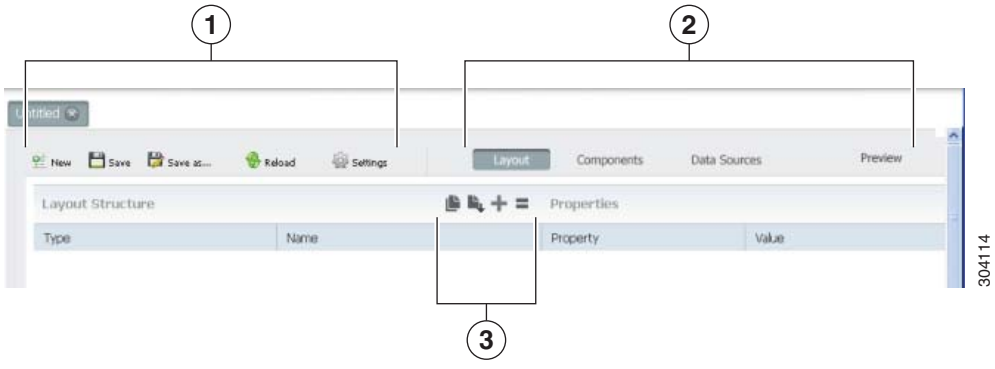
Navigating the Dashboard Workspace

To launch the dashboard workspace, from the File menu, choose **File > New > Dashboard**.

The dashboard workspace appears (Figure 5-1). The workspace has the following general components:

- Top level menu—Provides actions that affect the overall dashboard:
  - New—Creates a new dashboard.
  - Save—Saves the dashboard.
  - Save as—Allows you to save a dashboard under a new name or in a different directory.
  - Reload—Reloads the dashboard.
  - Settings—Displays dashboard settings. Settings include the dashboard title, author, and description. It also shows the dashboard style and type. Currently, only one style, Clean, and one dashboard type, blueprint, are available.
- Dashboard work Areas—All elements you work with to create your dashboard are contained in three work areas:
  - Layout—The work area where you define the dashboard layout.
  - Components—Contain all elements that you can add to the dashboard: charts, parameters, scripts, selectors, components, and other elements.
  - Data Sources—Define the sources of data used in the dashboard.
  - Preview—Allows you to preview your dashboard during its development.
- Toolbar—Provide common actions used during dashboard creation. The actions vary, depending on what work area you are working in. For example, in Layout, toolbar options are Save, Save as template, Add Resource, and Add Row. In Components and Data Sources, the toolbar options are Duplicate Component and Delete.

Figure 5-1 Dashboard Workspace



1	Top level menu	**3**	Work area toolbar
2	Work areas		

Creating a New Dashboard

In the following procedure, you will create a new dashboard and set up the dashboard layout.

To create a new dashboard:

Step 1 Log into the user console. For procedures, see Navigating the Dashboard Workspace, page 5-2.

Step 2 Review the folders under Browse. If you will store your dashboard in one of them, continue with the next step. If you need to create a new directory:

a. Right-click the Browse area and choose **New Folder**.

b. In the New Folder dialog, enter the new folder name, then click **OK**.

Step 3 From the File menu, choose **File > New > Dashboard.**

A new dashboard is created.

Step 4 From the File menu, choose **Save as**.

Step 5 In the Save as dialog box:

a. Choose the directory where you want to save your file. All dashboard files must be stored in a Sample subdirectory.

Note All dashboards are associated with the Prime Analytics installation. You cannot move or edit dashboard files manually because they require bipuser permission.

b. In the File Name field, enter the dashboard file name.

c. (Optional) In the Title and Description fields, enter the dashboard title and description.

d. Click **OK**.

Step 6 From the View menu, choose **Refresh**.

Your new directory is displayed in the directory where you placed it.

Step 7 Close the Untitled dashboard.

Step 8 Under Browse, select the folder containing your new dashboard.

Step 9 Under Files, select the new dashboard, then click the **Edit** tool.

Note If you choose Open, the blank dashboard will appear, but you will not be able to edit it.

Your dashboard appears in the work area with dashboard edit tools available. The dashboard tab name is Edit: *dashboardtitle*.

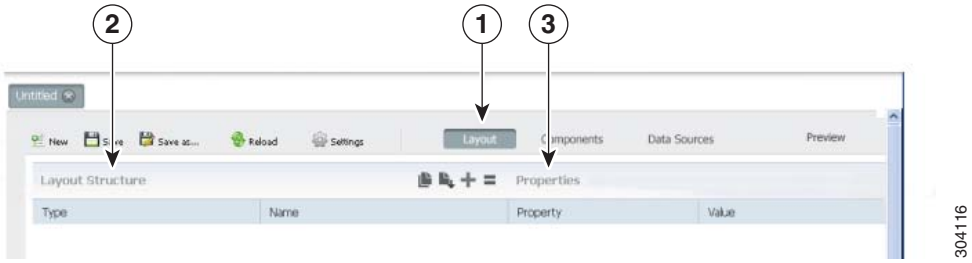
Step 10 Continue with the following topics:

- Setting Up the Dashboard Layout, page 5-4
- Adding Components to Dashboards, page 5-7
- Adding Data Source Queries to Dashboards, page 5-12

Setting Up the Dashboard Layout

When you first open a dashboard, the layout area is displayed with an empty layout (as shown in Figure 5-2). The layout workspace consists of two areas, Layout Structure and Properties. Layout Structure displays the current layout design. Properties displays the properties of the element currently is selected under Layout Structure.

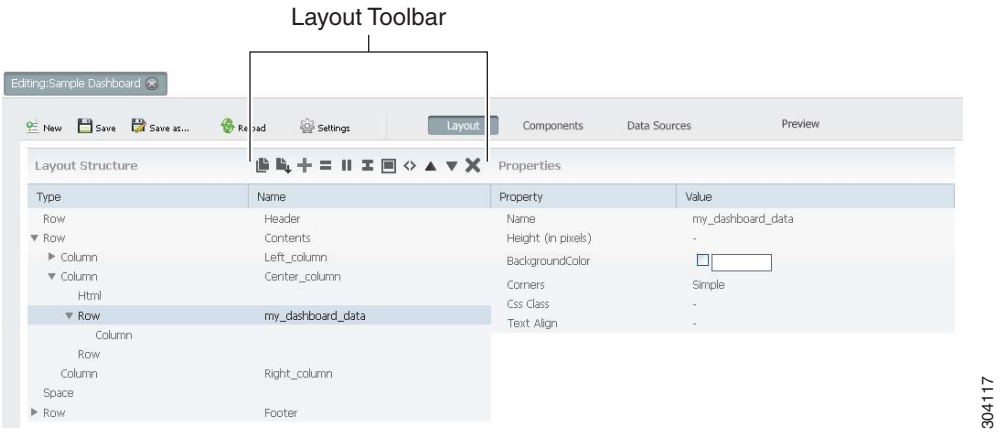
Figure 5-2 Layout Workspace



1	Layout workspace	**3**	Properties
2	Layout Structure		

Figure 5-3 shows the Layout workspace with a simple page design. The left pane shows that the page is divided in three rows labeled Header, Contents, and Footer. The Header row is a simple row containing no elements. The Contents row has several elements. These elements are expanded. You can expand or collapse elements by clicking the triangle located to left of the row element. In the example, the Footer row is collapsed. The triangle in front points to the right, indicating that this row contains collapsed elements.

Figure 5-3 Dashboard Workspace With Sample Layout



Layout Toolbar

Use the layout toolbar to customize your layout. The toolbar is dynamic, and displays between two to eleven tools, depending on the layout objects that you select. Actions that you can perform include:

- Save as template—Saves the dashboard as a template. The template includes the complete layout; you can choose whether the template also includes the components and data sources that are defined in the Components and Data Sources workspaces. After you save the dashboard as a template, a new empty dashboard is displayed.
- Apply template—Allows you to apply a template to the dashboard. Eight standard layouts are provided, including two, three, and four column templates, and five others.

Note The Apply Template tool is only available when you display a new template with no added elements. If you save a it as a template, you must open it as a normal dashboard file.

- Add resource—Adds a cascading style sheet (.css) or a JavaScript (.js) resource to the dashboard.
- Add row—Adds a row to the layout. If the currently selected element is a column, the row is added as a column element. Otherwise, the row is added at the same level as the selected element.
- Add column—Adds a column to the layout. If the current element is a row, the column is added as a column element. Otherwise, the column is added at the same level the selected element.
- Spacer—Add space between subsequent elements. If a row is selected, a vertical spacer is added. If a column is selected, a horizontal spacer is added.
- Add image—Adds an image to the current select row or column element. When you select this action, a dialog appears
- Add HTML—Adds an HTML block to the selected row or column element.
- Move up—Moves the current element one step up in the list (while remaining at the same level).
- Move down—Moves the current element one step down.
- Delete—Deletes the selected element.

Layout Properties

Layout properties also vary, depending on the element that you select. The first property in Figure 5-3 is Name. This property sets the element name. This name is shown in the Name column of the layout structure as well, and can also be used to attach components to the layout cells. You can use any alphanumeric character in a name. Underscores (_) are the only permissible special character in a name.

The dashboard layouts are based on the blueprint.css stylesheet. To minimize the need to manually adjust widths, blueprint.css divides a screen into 24 columns. If you create a column in the layout workspace it has three blueprint.css parameters:

- Span size—The column width.
- Prepend size—The amount of empty space to insert before the column.
- Append size—The amount of empty space to insert after the column.

The layout workspace width is set in blueprint.css columns. Although you can force the width of a component in pixels in the component view, it is not recommended. The recommended practice is to leave the component width empty so its width can be determined from its contents and the cell layout ( ) where the component is placed.

If the combined width of the columns on a line exceed the page width, the columns that do not fit are wrapped to the next line. The blueprint.css does not provide a height requirement. Column height is based on the height of its contents. Therefore, an empty column is invisible as it has a height of 0 pixels (unless you explicitly set the column height). For more information on the blueprint.css framework see http://www.blueprintcss.org.

Color Editor

Layouts include an option to set the background color property of rows and columns.

To set the background color:

Step 1 In the Background Color property, click the small box.

The color editor is displayed.

Step 2 Select the Hue (color range) by drawing a tiny rectangle on the rainbow selector in the middle to the appropriate color range,

Step 3 Select the saturation and brightness by drawing the white circle in the square box to the exact color you want (initially this circle is located in the left bottom of the box), and

Step 4 Click the color circle located in the lower right corner of the color editor to confirm the selection.

Note You can also enter the RGB, HSB or hexadecimal color codes in the R, G, B, or H, S, B, or # fields.

Working in the Layout Workspace

As you work with the dashboard editor Layout workspace, keep the following in mind:

- When entering a property, always press Enter to save your property. If you do not press Enter and select another layout element, the editor discards the value you entered without notice. Because you changed to a new layout element, you might not notice your data entry was discarded. If you use your mouse to select another property, the data entry for the preceding property is accepted.
- Some fields are completed automatically. These fields are identified with a drop-down list below the field.
- Use care when adding a multiple columns or rows simultaneously. As soon as you add an element the toolbar might change. For example if you start with an empty layout, the Add Row button is in the far right toolbar position. However, as soon as a row is added, the Delete tool added at the far right position of the toolbar. Double clicking the far right tool in an empty layout adds a row and immediately deletes it with the second click, resulting in an empty layout.
- The Properties area shows the properties of the layout currently-selected element. These properties vary by element, so the list changes dynamically if you move to another element in the layout structure.

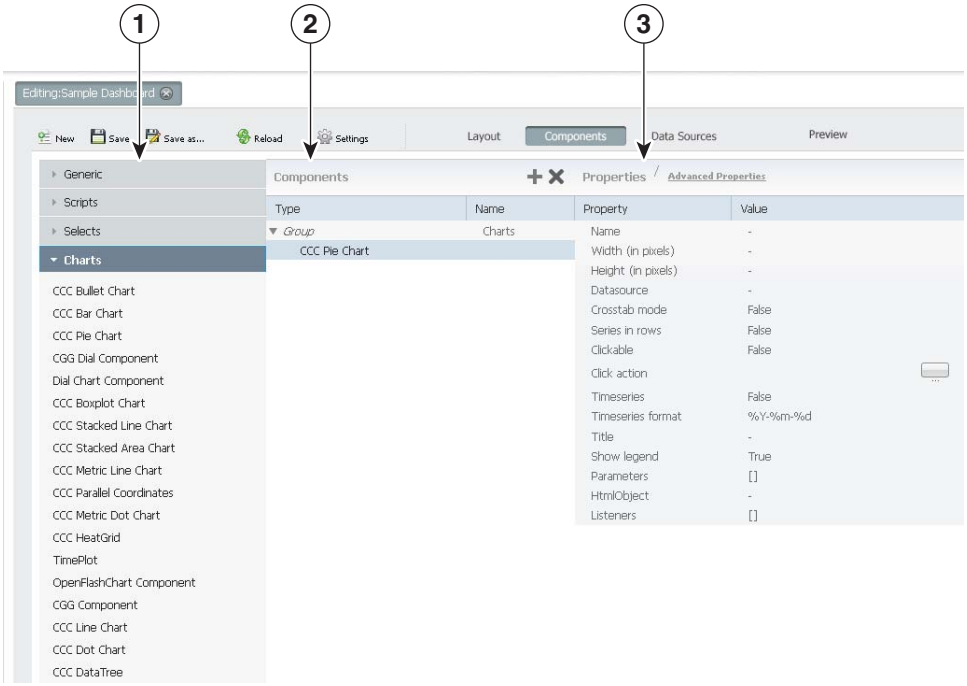
Adding Components to Dashboards

After you set up your dashboard layout (see Setting Up the Dashboard Layout, page 5-4), you can begin adding components to it. A wide range of components are available, from scripts and charts to buttons and maps. Familiarizing yourself components will take time. These topics will provide a general overview.

To add components, within a dashboard, choose **Components**.

The component workspace, shown in Figure 5-4, is displayed.

Figure 5-4 Components Workspace



1	Components list	**3**	Properties
2	Selected components		

This workspace has three areas. The left area displays components classes containing components that you can add to your dashboard. The middle area shows the components that have been added to the dashboard, and the right area shows the properties of the components selected in the Components list.

Component Categories Area

Available components are divided into eleven categories: Generic, Scripts, Selects (#1), Charts, Others, Widgets, Real Time Charts, Community Contributions, CDF Core Functionality, Custom, and Selects (#2). Many components are provided with the Pentaho Community Tools (CTools) application. Pentaho CTools acronyms referenced in the components include:

- CCC—Community Chart Components)
- CDE—Community Dashboard Editor

- CGG—Community Graphics Generator
- CDF—Community Dashboard Framework

For more information about Pentaho Community Tools, see http://community.pentaho.com.

Dashboard components provided with Prime Analytics include:

- Generic—Contains the following parameters:
  - Simple parameter—Adds a JavaScript parameter that sets a primitive default value, such as integer, double, string, or others. You can use JavaScript parameters with other parameters to connect components.
  - Custom parameter—Adds a JavaScript parameter that sets a compound-value default, for example, array, object, function. It can also be used when the value is initialized by another Java function.
  - Date parameter—Sets a date. This can be a specific date, for example today's date, or it can be linked to a date picker.
- Scripts—Adds a script to the dashboard. The provided JavaScript Function script defines a JavaScript function.

  Note If you have a long function or multiple functions to add, you can also place the JavaScript file on the server and include file as a layout workspace resource.

- Selects—Adds single-function elements:
  - MultiButtonComponent—Adds a multibutton component.
  - SelectMulti Component—Adds a multiple selection component. Users can choose one or more elements from a list and place them in a JavaScript variable.
  - TextInput Component—Text input without auto-completion.
  - Date Range Input Component—Adds a date range input. However, many servers do not permit this SQL query, as the client-side SQL introduces a security risk.
  - Date Input Component—Allows users to add a date.
  - Check Component—Lists one or more check boxes and connects them to a JavaScript variable.
  - Radiobutton Component—Adds a radio button that selects one item from a list of items.
  - TextareaInput Component—Adds an area where text can be entered.
  - Auto complete Component—Adds a text entry component with auto completion capability.
  - Month Picker Component—Adds a month selection element.
  - Simple Auto Complete Component—Adds a simple text entry component with auto completion capability.
  - Select Component—Presents a list box to select a parameter. It returns the selected parameter or its display value.
- Charts
  - CCC Bullet Chart
  - CCC Bar Chart—See CCC dot chart.
  - CCC Pie Chart—See CCC dot chart.
  - CGG Dial Component

  - Dial Chart component
  - CCC Boxplot chart—See CCC dot chart.
  - CCC Stacked Line Chart—See CCC dot chart.
  - CCC Stacked Area Chart—See CCC dot chart.
  - CCC Metric Line Chart—See CCC dot chart.
  - CCC Parallel Coordinates Chart—See CCC dot chart.
  - CCC Metric Doc Chart—See CCC dot chart.
  - CCC HeatGrid—See CCC dot chart.
  - TimePlot
  - OpenFlashChart component—Component using OpenFlashChart for visualization.
  - CGG Component
  - CCC Line chart—See CCC dot chart.
  - CCC Dot chart—the CCC charts are the preferred charts in the CDE because these charts look pleasant and have many features for interaction, like tooltips, animation and drill-down options.
  - CCC Data Tree
  - CCC Waterfall Chart
  - Chart component—Component using the Jfree chart graphics.
  - CCC 100% Stacked Bar Chart
  - Protovis component—A graphical toolkit for visualization. The CCC-chart are also based on Protovis. However, this components gives you access to the full power of the protovis library, which is a powerful javaScript library for a wide range of visualizations.
- Others—Provides a variety of additional components:
  - Freeform Component
  - Related Content Component
  - Pivot Link Component
  - Mobile Navigation Component
  - Xaction Component
  - Execute XAction Component
  - Export Button
  - Text Component—Shows the result of an expression, where the expression is provided by a JavaScript function.
  - Query Component— Executes a SQL or MDX query and returns the result.
  - PRPT Component—Executes the Pentaho Reporting unified format.
  - Table Component—Shows the contents of a data source in table format. By default, tables are sorted, searchable, and paginated. An extensive options list to customize the table component is provided.
  - Button Component
  - Comments Component—Adds a comments section to a page.
  - Popup

  - Traffic Component—Generates a traffic light image without needing an xaction file.
  - Duplicate Component
  - Execute Prpt Component—Executes a PRPT and displays the result in a new popup window.
  - Pivot Component—Executes a JPivot Action Sequence and creates an iframe where the pivot table is embedded.
  - ExportPopupComponent
  - Navigation Menu Component—Introduces a menu that allows you to browse the complete Pentaho solution repository (non-hidden folders).
- Widgets—Provides a single, general-use widget.
- Real Time Charts—Charts designed for Prime Analytics continuous query data streams.
  - Real Time Line Dual Y
  - Real Time Column
  - Real Time StackedArea
  - Real Time Line
  - Real Time Angular Gauge
- Community Contributions—Provides components provided by the Pentaho community. In this release, a Google Analytics component is provided.
- CDF Core Functionality—Provides the Pentaho CDF View Manager Component component:
- Custom—Provides a set of custom components:
  - Raphael component—Raphael is a library for Scalable Vector Graphics (svg).
  - NewMapComponent
  - AjaxRequestComponent
  - SiteMap
- Selects—Provides an additional OLAP Selector component.

Components Area

The selected components area shows the components that you added to the dashboard. These component are grouped by category. In the example in Figure 5-4, one component is added, a CCC Pie chart that belongs to the group charts. The middle pane has a dynamic toolbar. Two tools are shown: Duplicate Component and Delete. If more charts were added, the Move Up and Move Down tools are added.

Properties Area

The component properties area shows the properties of the selected component. Next graph shows component workspace. The available list of properties is already fairly long. For the CCC Pie chart in figure 6 this list contains 15 properties that can be set. If you click Advanced Properties, the advanced properties are added to the list. Properties that appear for many components include:

- Name—The component name. This name is also in the components list.
- Width—The preferred component width in pixels. By default the width is based on the layout you selected in the layout workspace. Only modify this option if the layout does not provide your component with sufficient space.
- Height—The component height in pixels. Only modify this parameter if the defaults do not provide sufficient room.

- Datasource—The data source defined in the Data Source workspace to which the component is connected.
- Parameter—The JavaScript parameters used by the component.
- HtmlObject—The name of the layout container defined in the Layout workspace that displays the component.
- Listeners—A list of JavaScript variables that this component watches. If the value of one of these variables changes, a redraw of the component is triggered.

Adding Charts to Dashboards

Below are the general steps associated with creating a chart:

1. Select a data source.
2. Build a query.
3. Set the data definitions: values, series, category.
4. Select a chart type and theme.
5. Enter labels for the chart title, and X and Y axes.
6. If applicable, adjust scaling and label rotation.
7. Place your chart in the dashboard.
8. Save your dashboard.

If you are new to charting, here are guidelines that may help you determine what type of chart is best suited for the data you want to present in your dashboard:

Bar Charts

Bar charts are useful when comparing items during a specific time period. Key words to think about when creating a bar chart are compare or rank. For example, if you want to compare items sold to show which one made the most profit, you can create a bar chart that ranks the products from the lowest to highest profit. The bar's length determines its ranking; the label identifies the item. Bar chart data can be presented horizontally or vertically depending on your requirements.

Pie Charts

Pie charts are useful when comparing parts of a whole. Key words associated with charts include, portion, share, and percentage. For example, if you want to demonstrate the proportion of the company's budget spent on health insurance, use a pie chart. To make the chart easier to read, limit the number of slices to five. Pie charts can also be exploded, which means certain slices are pulled away from the remainder of the chart for emphasis.

Line Charts

Line charts are useful for showing changes over time. Key words associated with data that is best suited for a line chart are trend, growth, and decline. If, for example, you want to show how product sales have changed over five years, use a line chart. The slope of the line helps users quickly identify the direction of the trend.

Dial Charts

Dial charts are often associated with Key Performance Indicators (KPIs). Dial charts are circular and contain a scale, a needle, and one or more a dial sectors. The dial sector is used to identify a specified area on a dial chart using a particular color. For example, you could have a dial plotting inventory with

a minimum dial value of 10000 and a maximum dial value of 50000. There could be a red dial sector for the region between 2000 and 4000 indicating that if the needle is in this area, there is a danger of a supply inventory shortage.

Area Charts

Area charts can be used to show a comparison of the same item during different points in time. Area charts are not designed to provide exact data; they provide users visual clues of the relative sizes of the items they represent.

Real Time Charts

Prime Analytics includes real time charts designed for continuous query data. These include:

- Real Time Line Dual Y
- Real Time Column
- Real Time Stacked Area
- Real Time Line
- Real Time Angular Gauge

Working with the Components Workspace

Most properties have an auto-completion function. When you edit such a property, a list pops up containing the available options. The list is context-sensitive. For example, if you click the HtmlObject, a list of all layout containers appears. However, if you click the parameter property, it will show a list of defined JavaScript variables. (This list will be limited to the JavaScript variables that you defined using the parameter components. It will not include parameters defined in your own JavaScript resources that you included in the dashboard editor.) When you enter a property value, the dashboard editor only does a basic validation. If you include a value that does not exist, the editor might not raise an exception. Also when you preview a dashboard, the editor might not call out errors. The only notification you might get is when your component is not drawn. Therefore, use the auto-completion function whenever possible. Most components have many properties. If a component fails, you might need considerable time to track down the property value that you misspelled.

The dashboard editor does not consistently warn you when you make an error in a component. The main error message is the failure of component render on the screen. Therefore:

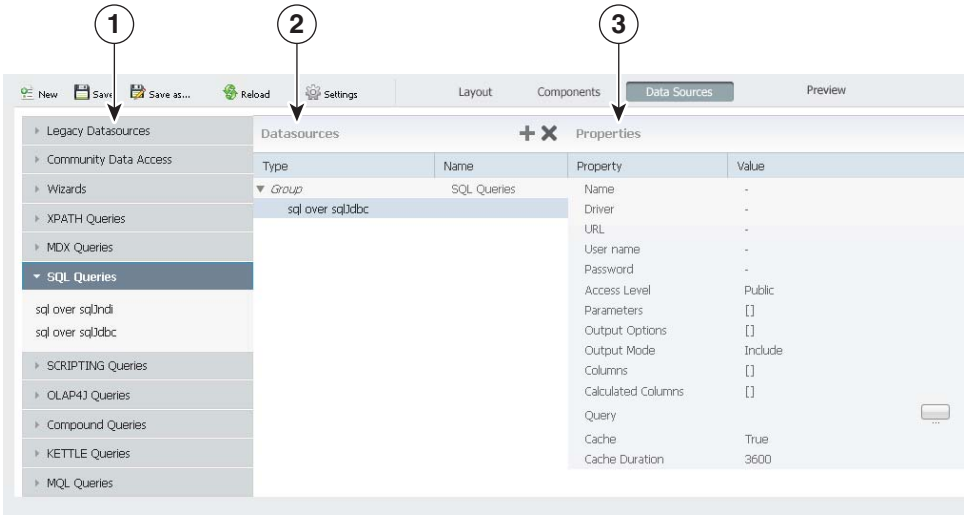
- Be very careful to avoid making errors.
- Preview your dashboard any time you make a significant change. Doing frequent previews takes little time, but can save you much time later because identifying the error is easier if you only made one or two small changes. If you save your file after each successful preview, you can easily backtrack to the previous version using the reload when you observe an error.

Adding Data Source Queries to Dashboards

The Data Sources workspace, shown in Figure 5-5, is where you add the data source queries to your dashboard components. The Data Sources workspace has the same layout as the Components workspace. Available data source queries are displayed in the left area, the middle area shows the data source query that you have created for the dashboard and the right area shows the properties of the current selected data source query.

Note The dashboard Data Source workspace is only used for queries on static databases. For continuous queries databases, see Adding Charts for Real-Time Data Sources, page 5-16.

Figure 5-5 Data Sources Workspace



1	Data source list	**3**	Properties
2	Selected data source		

Available Data Source Queries

The left area of the Data Sources workspace displays the data source queries and other processing that you can add to the dashboard. These include:

- Legacy Data Sources
  - OLAP MDX query
  - SQL query
  - Xaction result set
  - Kettle transformation
- Community Data Access
  - CDA data source—Includes an existing .cda file in your dashboard
- Wizards
  - OLAP members wizard
  - OLAP Chart wizard
- XPATH Queries—XML Path (XPath) queries. XPath is a query language used to select nodes from an XML document.
  - Xpath over Xpath
- MDX Queries—MultiDimensional eXpressions queries. MDX queries provide a specialized syntax for querying data in Online Analytical Processing cubes.

  - mdx over mondrianJndi—MDX over Mondrian JNDI.
  - denormalizedMdx over mondrianJdbc—Denormalized MDX over Mondrian JDBC.
  - denormalizedMdx over mondrianJndi——Denormalized MDX over Mondrian JNDI.
  - mdx over mondrianJdbc——MDX over Mondrian JDBC.
- SQL Queries
  - Sql over sqlJdbc—Access a database using the Java DataBase Connectivity API (JDBC).
  - Sql over sqlJndi—Access a database using the Java Naming and Directory Interface (JDNI).

Note Many data source queries are available in JDBC or JNDI. The JDBC API provides universal data access from the Java programming language. Using the JDBC API you can access virtually any data source, from relational databases to spreadsheets and flat files. JDBC also provides a common base on which tools and alternate interfaces can be built. JNDI is part of the Java platform. It provides applications based on Java technology with a unified interface to multiple naming and directory services.When using the JNDI data sources, define the data source in the Prime Analytics Administration Console. If you must access several tables from a database, use JDNI because it is referenced using a single parameter, whereas a JDBC data source requires four parameters (database name, connection-URL, username, and Password) to connect to the database server.

- Scripting Queries
  - Scriptable over scripting—Currently the only supporting scripting language is BeanShell, which is a small and embeddable Java source interpreter.
- OLAP4J Queries—OLAP4J queries. OLAP4J is a common API for any OLAP server.
  - Olap4J over olapJdbc—OLAP4J over OLAP JDBC.
  - Olap4J over olapJndi—OLAP4J over OLAP JNDI.
  - denormalizedOlap4j over olapJdbc—Denormalized OLAP4J over OLAP JDBC.
  - denormalizedOlap4j over olapJndi——Denormalized OLAP4J over OLAP JNDI
- Compound Queries
  - Join—Joins two data sources. This is often used to join tables residing in different databases.
  - Union—Concatenate two data sources that have the same column format.
- KETTLE Queries—Pentaho Data Integration Community Edition queries.
  - Kettle over kettleTransFromFile
- MQL Queries—Metaweb Query Language queries.
  - Mql over metadata

Data Source Area

The Data Source area in the middle of the workspace shows the current data sources defined for the dashboard. Above the data source list you have the same dynamic toolbar as described for the components workspace, allowing you to delete or duplicate data sources, or move them up or down.

Data Source Properties

The Data Source Properties shows the selected data source properties. Properties that appear for most data sources include:

- Name—The data source name. This name is also be shown in the middle area. It is the name used to make a connection to the data source from components in the Component workspace.
- Access Level—The data source access level.
- Parameters—JavaScript parameters that you can use to create a parameterized data sources

Note The parameterization is limited by the underlying data engine. In an SQL query you can use the parameter to limit the result set (WHERE statement) or to change grouping or ordering. However, in SQL you cannot parameterize the columns that you want to retrieve.

- Output Options—The data source output options.
- Output Mode—The data source output mode.
- Columns—The data source columns.
- Calculated Columns—The data source calculated columns.
- Query—The query used to extract data from data source.
- Cache———The data source cache.
- Cache Duration———The data source cache duration.

Working With the Data Source Workspace

The dashboard editor does not check your queries for errors. If your query has an error, the query usually fails. Errors returned by the database engine are not shown to you. The only feedback you receive is the failure of the component using the data to render. Therefore, the following practices are recommended:

- You copy and paste the query to your database/mdx client to test whether this query is valid and returns the correct result set.
- Use the CDA preview to verify the data source to see whether you get the proper results. An error in the data source settings might prevent the data source from returning the right result even when the query is correct. The CDA definitions are stored in a separate file (the .cda file). This file can be opened by clicking it in the user console URL field. However, there are two caveats:
  - Always save the CDE; otherwise the new definition is not be stored in the .cda file. It is stored in a temporary file that is not visible in the file browser.
  - If the data-source is executed, either in the CDA preview, or because you used the CDE preview, the results are stored in the CDA cache. The results are also cached if the query returns zero records due to an error in your query. Always save the dashboard to clear the CDA cache after you repair or modify a data source because some versions of the CDA do not use the temporary CDA file22. Clearing the CDA cache is needed in some CDE versions because the CDA does not detect that the .cda file is more recent than the cached results.

Adding Charts for Real-Time Data Sources

Complete the following steps to add data sources for real-time components:

Step 1 Log into the Prime Analytics user console as an administrator or designer user. See Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **New > Dashboard**.

Step 3 In the Layout workspace toolbar, click **Apply Template**.

Step 4 In the template dialog box, select a template and click **OK**.

Step 5 Click **Components**.

Step 6 Click **Real Time Charts** and choose one of the following charts:

- Real Time Line DY
- Real Time Column
- Real Time Stacked Area
- Real Time Line
- Real Time Angular Gauge

Step 7 For the chart you selected, enter the following minimum properties:

- Name
- Width
- Height
- HtmlObject.

Step 8 Next to Path, click **View SQL** to display a list of available continuous SQL connections.

Step 9 Click on **Path** and select the continuous SQL (CDA file).

Step 10 Enter **1** in DataAccess ID.

Step 11 On the dashboard toolbar, click **Save as**…" to save the dashboard in a location of your choosing.

Step 12 Click **Preview** to view the results.

Note The workflow for real-time charts differs from other components. The dashboard editor Data Sources workspace is not used for real-time components.

Improving Dashboard Performance

As you work with dashboards, you might find the need to improve performance in certain situations. The following provides some performance improve dashboard performance, For example, to retrieve records you use: select cq_close(*), count(*) from event <slices '5 seconds'>. Each time you get 2000 records. To make this more manageable, use an advanced property on the user console to set the maximum number of data points to plot. If you set it 20, it will show 20 records each time.

Because 2000 is much more than 20, a performance issue might arise. To solve this, add a limit to the SQL, for example, select cq_close(*), count(*) from event <slices '5 seconds'> limit 20. However, setting the limit to 20 means you cannot see 90% of the records. While no ideal solution exists, to see more data you can increase the maximum number of data points to plot, for example, 100. However, if this setting is too large, individual data points might be difficult to see on the dashboards. Ultimately, you will need to find a balance, so experiment to find the best balance.

Another performance improvement suggestion is to remove charts that aren't used. For example, you add a static chart, such as a pie chart, to a component. You might also add a JDBC data source component to the chart. You might later decide you don't want to see the chart show and remove the HtmlObject relation, such as Panel_1, from property while leaving the chart still in component.

The next time you run the dashboard, the chart will not be visible. However, it will still have a performance cost because of the JDBC data connection. Therefore, if you decide you do not want to display a chart, always delete it from the component to avoid drains on performance.

Creating Reports

The following topics provide an overview to Cisco Prime Analytics reports.

Overview to Report Creation

For reports, Prime Analytics has implemented Saiku Reports. See *http:/www.analytical-labs.com*. In general, Saiku Reports are intuitive and easy to create. The Prime Analytics report generation process includes the following steps:

1. Connect to a data source. This is usually a database, although you can also pull data from a flat file.
2. Constrain the data with a query.
3. Arrange the data elements in the report workspace
4. Apply report formatting; this can be through application of a report template or by formatting text and background color manually in your report.
5. Add formulas or calculated fields using data retrieved from your query.
6. Save your report in one of multiple formats including CSV, text, PDF, Excel, and many others.

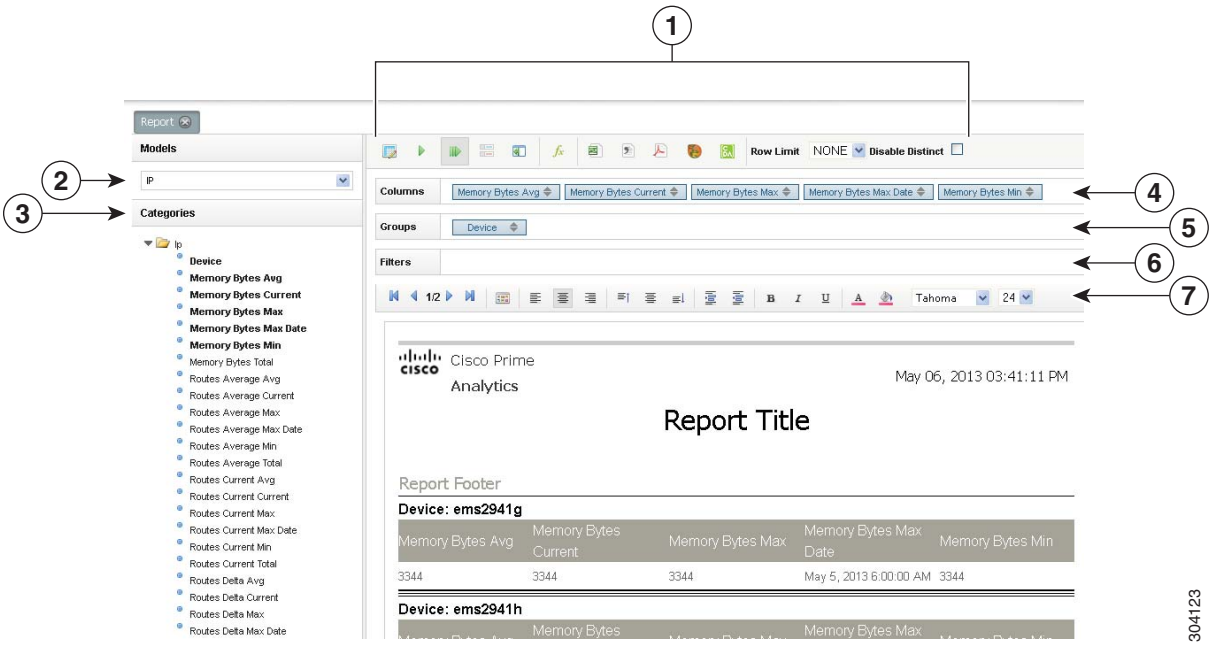
Your report will consist mostly of data retrieved from a database query that you will create through the data source wizard or by hand. After you have a dataset, you can further constrain it to show specific details, and then move on to report layout and design.

In general, you will find report creation fairly intuitive. Figure 6-1 shows the report workspace. Report workspace elements include:

- Report toolbar—Provides report actions including global display and report export options.
- Model—The data source model used for the report.
- Categories—The data source model categories you can choose to include in the report.
- Columns—The report columns, selected from the data source categories.
- Groups—Allows you to group your report based on a model category.
- Filter—Allows you to filter your report based on a model category.

- Report format toolbar—Provides options for formatting your report, including the ability to apply a report template.

Figure 6-1 Reports Workspace



1	Report toolbar	**5**	Groups
2	Models	**6**	Filters
3	Categories	**7**	Format toolbar
4	Columns		

Creating Reports

To create a report, complete the following steps:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage > Data Source.**

Step 3 Verify the data source you want to use for the report is displayed. If not, you must create it. For procedures, see Chapter 4, “Setting Up the Data Sources.”

Step 4 From the File menu, choose **New > Report.**

A new Report tab is displayed.

Step 5 Save your report:

a. From the File menu, choose **Save As**.

b. In the Filename field, enter the report name.

c. In the Location field, enter the directory where you want to store the report or double-click the directories to navigate to the directory you want.

d. Click **Save**.

Step 6 Under Models, choose your report data source.

The available report categories are displayed.

Step 7 Click the categories you want included in your report.

The categories are added to the Columns space and the category text is placed in bold text.

Step 8 Click a column.

Step 9 In the Setup Column *column_name* dialog box, modify any of the following parameters:

- Display Name—Edits the column name displayed in the report.
- Format—Adds text or numbers before or after the report data.
- Hide Repeating Values—If checked, hides repeating column values.
- Hide on Report—If checked, hides the column from display in the report. This parameter is useful if you want to prepare multiple report versions and temporarily hide columns.
- Aggregation—Sets the aggregation method used. The methods displayed depend on the column type. For numbers, the following aggregation methods are available:
  - Count Distinct
  - Maximum
  - Average
  - Sum
  - Count
  - Minimum
- Summary—Adds a summary cell to the column and defines the summary method, which are the same as listed for Aggregation.

Step 10 After you complete the column edits, click **Save**.

Step 11 Repeat Steps 8 through 10 for each report column.

Step 12 If you want to group your columns, drag the category you want to use to the Group space.

For example, if you want to group columns by device, you would move the Device column to the Group space. The report will change to reflect your group

Step 13 If needed, repeat Steps 8 through 10 to modify parameters of the category column.

Step 14 If you want to filter your report, drag the category you want to use to the Filter space.

Step 15 If needed, repeat Steps 8 through 10 to modify parameters of the filter column.

Step 16 If needed, add a calculated column to your report:

a. On the report toolbar, click **Add Calculated Column**.

b. Enter the column name and the formula you want to use.

c. As needed, complete other column parameters listed in Step 9.

d. Click **Save**.

After you add your report columns and set up any groups, filters, and calculated columns, you can begin formatting the report.

Step 17 Double-click the Report Title text and type the name of your report.

Step 18 As needed, modify the title format using the format toolbar:

- Horizontal Alignment: left, centered, right
- Vertical Alignment: top, middle, bottom
- Padding: right or left
- Text: bold, italic, underlining, color, background color, font type, and size.

Note All format changes except text and background color are applied immediately. To modify the text and background color, click the Text or Background Color tool, choose the color on the color panel, then click the small circle at the lower bottom of the color panel to apply the color.

Step 19 Double-click Report Footer and repeat Step 17 through 18 to format the report footer.

You can format the report body in one of two ways:

- You can highlight individual table cells, rows, or columns and use the format toolbar to apply the formatting, or,
- You can apply a template and global report settings.

Step 20 To apply a template your report,

a. On the report format toolbar, click **Select Template**.

b. Under Document Template, scroll through the templates and select the one you want to apply. By default, the following templates are provided:

- Saiku_small
- jade_4_left_alignment_grid
- cobalt_4_left_alignment_grid
- maple_4_left_alignment_grid
- nickel_4_left_alignment_grid

c. Under Page Settings, choose the page setting you want to apply, and indicate whether you want vertical or horizontal orientation. Available page settings are based on ANSI, ISO, and other standards organizations.

d. Under Page Margins, modify the page margins, if needed. The default is 18 for all margins.

e. Click **Save**.

Your template and global page settings are applied.

Step 21 If needed, you limit the number of rows that are displayed by clicking Row Limit and choosing one of the following row limits: None, 10, 100, 1000.

Step 22 After completing your report setup, from the File menu, choose **Save**.

Editing Reports

Editing reports follows much the same process as report creation. To edit a report:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage** > **Edit**.

Step 3 In the Open dialog box, navigate to the directory containing the report you want to edit, select it, then click **Open**.

The report is displayed with the tab name: Editing *report_name*.

Step 4 Complete Steps 8 through 21 of the "Creating Reports" procedure on page 6-2 to edit any report items and layout.

Step 5 When finished, from the File menu, choose **Save**.

Exporting Reports

You can export reports into multiple formats. To export a report:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage** > **Edit**.

Step 3 In the Open dialog box, navigate to the directory containing the report you want to export, select it, then click **Open**.

The report is displayed with the tab name: Editing *report_name*.

Step 4 On the report toolbar, click one of the following options:

- **Export XLS**—Saves your report in Excel format to a directory of your choosing.
- **Export CSV**—Saves your report in CSV format to a directory of your choosing.
- **Export PDF**—Saves your report in PDF format to a directory of your choosing.
- **Export CSV**—Saves your report in Excel format to a directory of your choosing.
- **Export PRPT**—Saves your report in Saiku report format to the Prime Analytics project directory of your choosing. A username and password is required.
- **Export CDA**—Saves your report in Pentaho Community Database Access format to the Prime Analytics project directory of your choosing.

Deleting Reports

To delete a report:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 In the Browse directory, navigate to the directory containing the report you want to delete.

Step 3 Choose the report, click the Options tool above the report file name and choose **Delete**.

Step 4 On the confirmation, click **OK**.

Creating an Analysis

Prime Analytics helps you visualize data trends and reveal useful information about your network by creating static reports from an analysis data source, traversing an analytics cube through an analytics report, showing how data points compare by using charts, and monitoring the status of certain trends and thresholds with dashboards.

The following topics tell you how to create an analysis in Cisco Prime Analytics:

Overview to Analytics

Prime Analytics analytics is based on Mondrian, an Online Analytical Processing (OLAP) server that enables you to analyze large data volumes in real-time. Mondrian queries are written in the Multidimensional Expressions (MDX) language. MDX queries data contained in a relational database (RDBMS) and presents results in multidimensional formats called cubes.

While relational databases store data in rows and columns, multidimensional datasets consist of axes and cells. For example, in Figure 7-1, the rows axis consists of all Product members: Books, Fiction, Non-fiction, and so forth. The columns axis consists of the cartesian products 2000 and 2001, and the calculation, Growth, plus the measures Unit sales and Dollar sales. Each cell represents sales of a product category in a particular year; for example, the dollar sales of Magazines in 2001 were $2,426.

Multidimensional dataset members are not always relational column values. Total, Books and Fiction are members at successive levels in a hierarchy; each is rolled up to the next. Even though Growth is at the same level as the years, 2000 and 2001, Growth is a calculated member, which introduces a formula for computing cells from other cells.

Figure 7-1 OLAP Dataset

Year		2000		2001		Growth
Product	Dollar sales	Unit sales	Dollar sales	Unit sales	Dollar sales	Unit sales
Total	$7,073	2,693	$7,636	3,008	8%	12%
— Books	$2,753	824	$3,331	966	21%	17%
—— Fiction	$1,341	424	$1,202	380	-10%	-10%
—— Non-fiction	$1,412	400	$2,129	586	51%	47%
— Magazines	$2,753	824	$2,426	766	-12%	-7%
— Greetings cards	$1,567	1,045	$1,879	1,276	20%	22%

304119

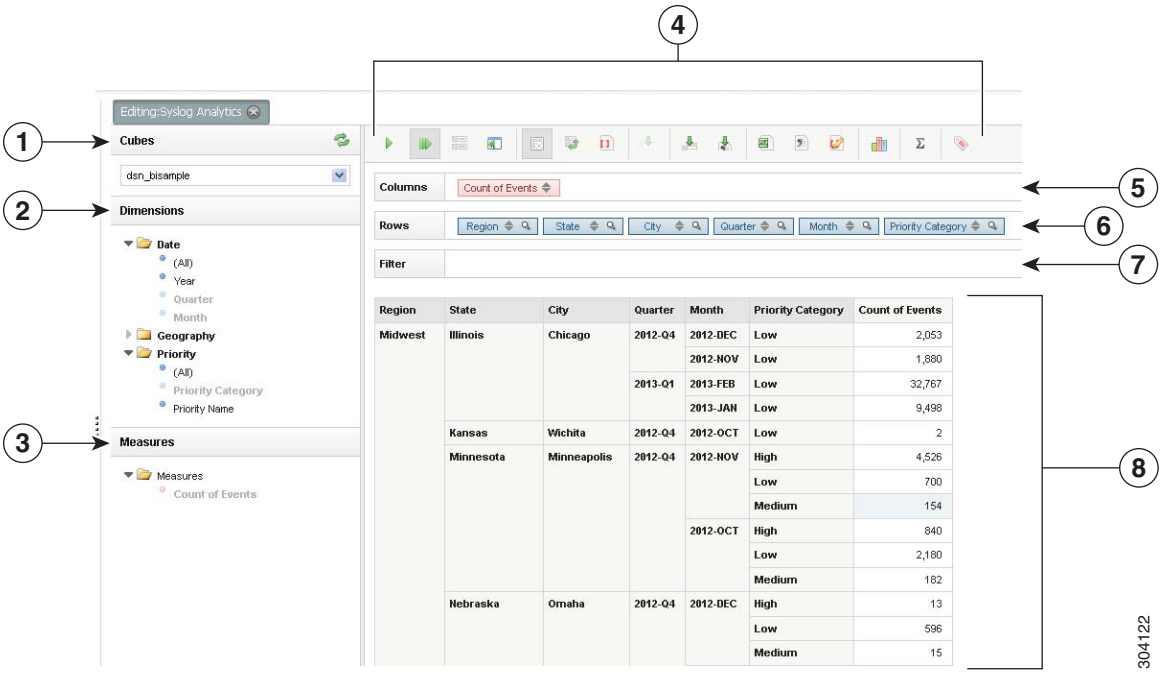
The dimensions used in this example—products, time, and measures—are three of many dimensions by which you can categorize and filter datasets. The collection of dimensions, hierarchies and measures is called a cube.

To prepare an analysis, you consolidate data from multiple sources into one source and optimize it for the metrics you want to analyze. You then create an analysis schema to describe the data. You improve that schema so that it meets your users needs; and create aggregation tables for frequently computed views.

Prime Analytics has implemented Saiku analytics to display your analytics data. Figure 7-2 shows the Analytics workspace with the sample application, Syslog Analytics, displayed. Workspace elements include:

- Cube—The multidimensional data source selected for analytics display.
- Dimensions—The data sets comprising the cube. Each dimension is an individual, non-overlapping data element. In the example, dimensions are data, geography, and priority.
- Measures—Numeric values that summarize various aggregation levels. In the example, the Count of Events is the measure.
- Analytics toolbar—Provides actions that allow you to manage the analysis, including different ways to display the analysis and export the data to other formats.
- Columns—The columns of the cube.
- Rows—The rows of the cube.
- Filter—The attribute used to filter the data.
- Analytics work area—The display of the cube dimensions, measures, columns, and rows.

Figure 7-2 OLAP Dataset



1	Cubes	**5**	Columns
2	Dimensions	**6**	Rows
3	Measures	**7**	Analytics data
4	Analytics toolbar	**8**	Analytics work area

Creating an Analysis

To create an analysis, complete the following steps:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage > Data Source.**

Step 3 Verify the cube you want to use for the report is displayed. If not, you must create it. For procedures, see Chapter 4, "Setting Up the Data Sources."

Step 4 From the File menu, choose **New > Analysis.**

A new Analytics tab is displayed.

Step 5 Save your analysis:

a. From the File menu, choose **Save As**.

b. In the Filename field, enter the analysis name.

c. In the Location field, enter the directory where you want to store the analysis or double-click the directories to navigate to the directory you want.

d. Click **Save**.

Step 6 Under Cubes, choose the cube you want to use as the analysis data source.

The available analytics dimensions and measures are displayed.

Step 7 Click the dimensions you want included in your analysis.

The dimensions are added to the Columns and Rows spaces and the dimension text is placed in faint text.

Step 8 Continue adding dimensions and measures to your analysis. If you want to filter your report, drag the dimension you want to use to the Filter space.

Note The analysis must have a minimum of one measure.

Step 9 As needed, Analytics toolbar actions to modify your analysis and its display:

- Run Query—Runs the analysis query manually.
- Automatic Query—If selected, automatically runs the query. This tool is a toggle; click to turn on, click to turn off.
- Toggle Fields—Displays or hides the Columns, Rows, and Filter spaces.
- Toggle Sidebar—Displays or hides the Cubes, Dimensions, and Measures tree.
- Swap Axis—Rotates the analysis axis.
- Show MDX—Displays the analysis MDX code in a popup window.
- Query Scenario—Is not available in this release.
- Cell drill-through options allow you to drill down a cell containing an analytics measure. Options include:
  - Cell Drill-Through on Cell—Allows you to drill through on individual analytics cells.
  - Export Drill-Through on Cell to CSV—Allows you to drill through on individual cells to the CSV.
- Export XLS—Exports the analysis as Excel file.
- Export CSV—Exports the analysis in CSV format.
- Switch to MDX Mode—Displays the MDX code in an area above the analysis.
- Basic Statistics—Displays the analysis minimum, maximum, sum, average, and standard deviation while hiding other analysis data.
- Tags—Allows you to add tags to individual cells. Select a cell, click Tag, Enter the tag text, then click Save. Clicking the tab displays the cell data. To remove the tag, click the small x just above the tag.
- Toggle Chart—Switches the analysis to graphic display. Five visual representations are available:
  - Bar
  - Stacked Bar
  - Line
  - Pie
  - Heat Grid

Step 10 After completing your analysis, from the File menu, choose **Save**.

Editing an Analysis

Editing an analysis follows much the same process as analysis creation. To edit an analysis:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage** > **Edit**.

Step 3 In the Open dialog box, navigate to the directory containing the analysis you want to edit, select it, then click **Open**.

The analysis is displayed with the tab name: Editing *analysis_name*.

Step 4 Complete Steps 6 through 10 of the "Creating an Analysis" procedure on page 7-3 to edit any analysis items and layout.

Step 5 When finished, from the File menu, choose **Save**.

Exporting an Analysis

You can export your analysis into an Excel spreadsheet or as a text file in CSV format. To export an analysis:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 From the File menu, choose **Manage** > **Edit**.

Step 3 In the Open dialog box, navigate to the directory containing the analysis you want to export, select it, then click **Open**.

The analysis is displayed with the tab name: Editing *analysis_name*.

Step 4 On the Analytics toolbar, click one of the following options:

- **Export XLS**—Saves your analysis in Excel format to a directory of your choosing.
- **Export CSV**—Saves your analysis in CSV format to a directory of your choosing.

Deleting an Analysis

To delete an analysis:

Step 1 Log into the Prime Analytics user console. For procedures, see Logging Into the User Console, page 3-1.

Step 2 In the Browse directory, navigate to the directory containing the analysis you want to delete.

Step 3 Choose the analysis, click the Options tool above the analysis file name and choose **Delete**.

Step 4 On the confirmation, click **OK**.

Managing Prime Analytics

Prime Analytics administrators perform many tasks that maintain the health and performance of the Prime Analytics BI platform and databases. Using the BI platform Administration Console, administrators can manage users and user roles, database connections, and database query schedules. Additional Administration functions include backing up the Prime Analytics BI platform and database and, if needed, restoring the BI platform and database.

Administrative functions and tasks are covered in the following topics:

Logging Into the BI Platform Administration Console

Accessing the Prime Analytics Administration console requires Admin security access. To log into the BI Administration Console:

Step 1 Launch your web browser, then enter one of the following in the browser URL field:

- ***<Prime Analytics BI Server>*:8080/pentaho/Admin**—Launches the Administration Console directly.
- ***<Prime Analytics BI Server>*:8080**—Displays the User Console. On the User Console under Administration, click **Settings**.

Step 2 At the login window, enter the Admin username and password, then click **OK**.

The Prime Analytics Administration Console appears. The console is divided into two areas:

- Admin Services—Includes the following administrative services:
  - Clean Repository—Removes files from the content repository that are more than 180 days old.
  - Scheduler Admin—Allows to manage the scheduler. See "Managing BI Server Schedules" procedure on page 8-10.
  - Manage Users and Roles—Allows you to manages Prime Analytics BI platform users and roles. (see Managing Users, page 8-2), database connections, (see Managing the Database Connections, page 8-7), services (see Managing BI Server Services, page 8-9, and schedules (see Managing BI Server Schedules, page 8-10).
  - Permissions—Allows you to set execute and overwrite permissions for content for users and roles.
  - Subscription Admin—Allows you to manage subscription content and schedules.
- Refresh—Includes the following items:
  - Plugin Adapter—Refreshes all the BI platform plugin applications.
  - Subscription publisher—Publishes imported schedules and content.
  - Update solution repository—Reads all of the solution files and regenerates the repository index.
  - Reporting Metadata—Refreshes the metadata used for ad hoc reporting.
  - Global Actions—Executes all global system actions defined in pentaho.xml.

Managing Users

BI platform user management tasks are performed using the Prime Analytics Administration Console > Manage Users and Roles > Users & Roles tab.

By default, the following users roles are provided with the Prime Analytics installation:

- Admin—Has full create, read, update, and delete access to all user and administrator functions including user and user role management.
- Designer—Has full create, read, update, and delete access to all User Console features including access the data source functionality. Cannot access the Administrator Console.
- Developer—Has full or partial create, read, update, and delete access for dashboards and roles assigned by the administrator. Cannot access the Administration Console and cannot access or edit data source functionality.
- Authenticated—A user who has logged in.
- Anonymous—A user who has not logged in.

In addition to default user roles, Table 8-1 lists the users provided with Prime Analytics by default.

Table 8-1 Default Users

Username	Authority
joe	Authenticated, Admin
admin	User required to manage users and user roles.

Table 8-1 Default Users (continued)

Username	Authority
katy	Authenticated, Designer
bill	Authenticated, Developer
sam	Authenticated

BI platform user management is covered in the following topics:

Changing Default User Passwords

By default, Prime Analytics ships with the following user passwords:

- joe/password
- admin/password
- katy/password
- bill/password
- sam/password

Changing the default passwords is something you should do following installation. Changing the password for users joe, katy, bill, and sam is performed by completing the "Editing BI Platform User Information" procedure on page 8-5.

The admin user is required to access the Manage Users and Roles window. To change the default admin password, complete the following steps:

Step 1 Login as root users.

Step 2 Stop the BI platform:

```
service biplatform stop
```

Step 3 Login as the bipuser.

Step 4 Change to the administration-console directory:

```
cd $PA_HOME/biplatform/administration-console
```

Step 5 Enter the new password:

```
java -cp lib/jetty-6.1.2.jar:lib/jetty-util-6.1.9.jar org.mortbay.jetty.security.Password
<user> <newpassword>
```

for example,

```
$ java -cp lib/jetty-6.1.2.jar:lib/jetty-util-6.1.9.jar
org.mortbay.jetty.security.Password admin newpassword
newpassword
newpassword
OBF:1uo91vn61ymf1yt41v1p1ym71v2p1yti1ylz1vnw1unp
MD5:5e9d11a14ad1c8dd77e98ef9b53fd1ba
CRYPT:adx.wc8vu/gJU
```

Step 6 Change to the config directory:

```
cd resource/config
```

Step 7 In login.properties, enter the new OBF entry:

admin: OBF:1uo91vn61ymf1yt41v1p1ym71v2p1yti1ylz1vnw1unp,admin

Step 8 Log in as the root user and start the BI platform:

```
service biplatform start
```

Creating New BI Platform Users

To create new BI platform users:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click the **Users** icon.

Step 5 Next to the Users list title, click **+** (Add User).

Step 6 In the Add User dialog box, enter the following:

- User Name
- Password
- Password Confirmation
- Description

Step 7 Click **OK**.

The new user is added to the Users list.

Step 8 Select the user you just created.

Step 9 Next to Assigned Roles, click **+** (Assign Role).

Step 10 In the Assign Role dialog box, choose the roles you want to assign to the user from the Available Roles list.

Step 11 Click **OK**.

Editing BI Platform User Information

To edit BI platform user information:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click the **Users** icon.

Step 5 Under Users, choose the user you want to edit.

Step 6 Under Details, edit any of the following information:

- User Name
- Password
- Password Confirmation
- Description
- Assigned Roles—Click **Assign Roles** or **Unassign Roles** to add or remove roles to and from the user.

Step 7 Click **OK**.

Tip If you have many users, you can find a specific user by entering the first few letters of the user's name in the Filter box. A list of names matching your entry appears.

Deleting BI Platform Users

To delete BI platform users:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click the **Users** icon.

Step 5 Under Users, choose the user you want to delete.

Step 6 Click **X** (Delete Users).

Step 7 On the confirmation, click **OK**.

Creating New BI Platform User Roles

To create new BI platform user role:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click **Roles**.

Step 5 Next to the Roles title, click **+** (Add Role).

Step 6 In the Add Role dialog box, enter the following:

- Role Name
- Description

Step 7 Click **OK**.

The new role is added to the Roles list.

Editing BI Platform User Roles

To edit a BI platform user role:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click **Roles**.

Step 5 Under Roles, choose the role you want to edit.

Step 6 Under Details, edit any of the following information:

- Role Name
- Description

Step 7 Click **OK**.

Tip If you have many roles, you can find a specific role by entering the first few letters of the role name in the Filter box. A list of roles matching your entry appears.

Deleting BI Platform User Roles

To delete a BI platform user role:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 On the Users & Roles tab, click **Roles**.

Step 5 Under Roles, choose the role you want to delete.

Step 6 Next to Assigned Roles, click **X** (Delete Roles).

Step 7 On the confirmation, click **OK**.

Managing the Database Connections

The BI platform Administration Console allows you to manage all connections to Prime Analytics databases, including access drivers and URLs, usernames and password required for access, maximum active connections, number of idle connections, validation queries, and other properties. Database connection management tasks are covered in the following topics:

- Creating a New Database Connection, page 8-7
- Editing Database Connections, page 8-8
- Deleting Database Connections, page 8-9

Creating a New Database Connection

To create a new database connection:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 Click the **Database Connections** tab.

Step 5 Next to Database Connections, click **+** (Add Database Connection).

Step 6 In the Add Database Connection dialog box, enter the following:

- Name—Enter a name for the database connection.
- Driver Class—Enter the database driver class. The database driver name you select depends on the type of database you are accessing. For example, org.hsqldb.jdbcDriver is a sample driver name for a HypersonicSQL database. The default driver is com.primeanalytics.Driver. Other drivers can be added using the "Installing the JDBC Drivers" procedure on page 4-1.
- User Name—Enter the username required to use the database

- Password—Enter the username password.
- URL—Enter the URL used to access the database. This is the URL of your database. For example, jdbc:hsqldb:hsql://localhost/sampledata. JDBC establishes a connection to a SQL-based database and sends and processes SQL statements.

Step 7 Click Test to test your connection. If the test is successful, continue with the next step. If not, verify the information in the previous step and test again.

Step 8 Click the **Advanced** icon.

Step 9 As needed, enter the following information

- Maximum Active Connections—The maximum number of active database connections that can be allocated from this pool at the same time.
- Number of Idle Connections—The maximum number of that can sit idle in this pool at the same time.
- Validation Query—The SQL query that can be used by the pool to validate connections before they are returned to the application. If specified, this query must be an SQL SELECT statement that returns at least one row.
- Wait—The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception.

Note If any fields are left empty, the BI platform uses the default values stored in the pentaho.xml.

Step 10 Click **OK**.

The new database connection is added to the Database Connections list.

Editing Database Connections

To edit BI platform database connections:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 Click the **Database Connections** tab.

Step 5 Under Database Connections, choose the database connection you want to edit.

Step 6 Edit any of the following database properties. For property descriptions, see Creating a New Database Connection, page 8-7.

- Name
- Driver Class
- User Name
- Password
- URL

Step 7 Click the **Advanced** icon.

Step 8 As needed, edit the following information

- Maximum Active Connections
- Number of Idle Connections
- Validation Query
- Wait

Step 9 Click **Test** to verify that your edits have not invalidated the connection. If the test is successful, continue with the next step. If not, verify the information you entered and test again.

Deleting Database Connections

To delete BI platform database connections:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 Click the **Database Connections** tab.

Step 5 Under Database Connections, choose the database connection you want to delete.

Step 6 Next to Database Connections, click **X** (Delete Database Connection).

Step 7 On the confirmation, click **OK**.

Managing BI Server Services

The Administration Console Services tab allows you to manage schedules and refresh the BI Server settings.

To manage BI platform server settings:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 Click the **Services** tab.

Perform any of the following services:

- Schedule—Schedules the daily removal of files created in the content repository located in /pentaho-solution/system/content that are over 180 days old. To change the number of days, edit the solution file clean_repository.xaction located in /pentaho-solution/admin. To change the recurrence, edit the solution file schedule-clean.xaction located in /pentaho-solution/admin.

- Execute—Removes files created in the content repository located in /pentaho-solution/system/content that are over 180 days old. To change, the number of days, edit the solution file clean_repository.xaction located in /pentaho-solution/admin.
- Refresh—Reads all solution files and regenerates the repository index.
- Reset Permissions—Resets all permissions.

Managing BI Server Schedules

The Administration Console Scheduler tab allows you to create, update, delete, run, suspend, and resume one or more private and public schedules, You can also suspend and resume the BI server scheduler itself. In the context of the BI platform, a schedule is a time (or group of times) associated with an action sequence (or group of action sequences). In many instances, the output of an action sequence associated with a public schedule is a report, for example, a device domain report to which a network administrator can subscribe. As the administrator, the schedule (or schedules) you designate determines when the scheduler allows the action sequence to run. Regular schedules are ad hoc, non-subscription schedules, which are associated with one action sequence only.

In addition to associating a time (or group of times in the case of a repeating schedule) with an action sequence or group of action sequences, the public schedule is also associated with a user's My Workspace. When an action sequence runs on its defined schedule, the output of the action sequence (typically a report) is archived in the My Workspace of the user(s) who have subscribed to that action sequence. This allows the subscribers to view the output of the action sequence (the report) at any time following its execution.

Management of server schedules is covered in the following topics:

- Creating BI Server Schedules, page 8-10
- Managing Schedules, page 8-11

Creating BI Server Schedules

You create BI server schedules associated with your action sequences using the Schedule Creator. The Schedule Creator allows to enter schedules without having to know cron syntax, although you can enter cron expressions in the Scheduler Creator if you prefer.

To create a BI server schedule:

Step 1 Log into the Prime Analytics Administration Console. See Logging Into the BI Platform Administration Console, page 8-1.

Step 2 Under Admin Services, click **Manage Users and Roles**.

Step 3 At the login screen, enter the user, admin, and the admin user password.

Step 4 Click the **Scheduler** tab.

Step 5 In the Schedule Creator dialog box Schedule tab, enter the following information:

- Public Schedule—Check if you want users to be able to access the schedule.
- Name—Enter a schedule name, for example, Monthly Device Availability Report.
- Group—Enter a group associated with the schedule, for example, Western Region.

- Description—Enter a description of the schedule. for example, Schedule runs on the first of each month, or Schedule runs on Monday of each week.
- Recurrence Type—Enter the frequency at which you want the action sequence to occur. You can:
  - Schedule the sequence to run once at a particular date and time,
  - Schedule the sequence to recur at regular intervals. The intervals can be expressed in seconds, minutes, hours, days, weeks, months, year.
  - Enter a cron string.

  The options change depending on the type of recurrence you select.

Step 6 Click **OK**.

Step 7 Click the **Selected Files** tab.

Step 8 Click the + (Select Item) icon.

Step 9 In the Select dialog box, navigate to the directory and select file(s) on which you want the action sequence to occur.

- To open a directory, choose the directory and click **Select**.
- To select a file(s), choose the file(s) and click **OK**.

Step 10 Click OK to close the Schedule Creator dialog box.

Managing Schedules

As you create new schedules, the schedules are added to the Scheduler tab list box. The schedule list provides the following information for each schedule:

- Name—The schedule name.
- Group—The group assigned to the schedule
- State—The schedule state, either Normal or Suspended.
- Next Fire Time—The next date and time the schedule will run.
- Last Fire Time—The last date and time the schedule ran.

Use the Schedule toolbar controls, listed in Table 8-2, to perform schedule management actions.

Table 8-2 Schedule Actions

Action	Description
Create	Creates a new schedule
Edit	Allows you to edit the schedule details.
Delete	Deletes a specified schedule if it is not currently executing. If the schedule is currently executing, it continues to execute but no new instances are run.
Suspend	Pauses a specified schedule. After the job is suspended, you must click Resume to start it again.
Resume	Resumes a previously suspended schedule. After the schedule is resumed the scheduler applies misfire rules, if needed.
Run	Runs a schedule immediately.

Table 8-2 Schedule Actions (continued)

Action	Description
Refresh	Refreshes the list of schedules.
Filter by	Allows you to search for a specific schedule by group name.

Enabling SSL on the BI Platform

The following procedures tell you how to enable Secure Socket Layer (SSL) to enhance security and enable encrypted communications between Prime Analytics client web browsers and the BI platform server, including the administration and user consoles. Complete the following procedures in order:

- Creating an SSL Certificate, page 8-12
- Installing the Certificate, page 8-13
- Configuring Tomcat, page 8-14
- Configuring the Jetty Server with the SSL Certificate, page 8-14
- Configuring the Prime Analytics Properties File, page 8-15
- Validating SSL, page 8-16

Creating an SSL Certificate

The following steps tell you how to create a self-signed SSL certificate using the keytool that comes with the Java Development Kit. (In a production environment, obtain a certificate from a trusted certification authority.)

To generate an SSL certificate:

Step 1 Log into the Prime Analytics BI platform server as the root user.

Step 2 Stop the BI platform server:

```
service biplatform stop
```

Step 3 Switch to the bipuser:

```
su - bipuser
```

Step 4 Navigate to bipuser home directory:

```
cd $HOME
```

Step 5 Generate the SSL key:

```
keytool -genkey -alias tomcat -keyalg RSA
```

Enter the following parameters:

- Enter keystore password—Enter **changeit.**
- Re-enter new password—Enter **changeit.**
- What is your first and last name?—Enter **localhost**. If you do not enter localhost, the HostnameVerifier will fail.

- Enter key password for <tomcat>—Press **Enter/Return** to use same password as the one entered for keystore.

For example:

```
keytool -genkey -alias tomcat -keyalg RSA
Enter keystore password: changeit
Re-enter new password: changeit
What is your first and last name?
  [Unknown]:  localhost
What is the name of your organizational unit?
  [Unknown]:
What is the name of your organization?
  [Unknown]:
What is the name of your City or Locality?
  [Unknown]:
What is the name of your State or Province?
  [Unknown]:
What is the two-letter country code for this unit?
  [Unknown]:
Is CN=localhost, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=Unknown correct?
  [no]:  yes
Enter key password for <tomcat>
        (RETURN if same as keystore password):changeit
```

Step 6 Continue with the "Installing the Certificate" procedure on page 8-13.

Installing the Certificate

Complete the following steps to install the certificate created in Creating an SSL Certificate, page 8-12.

Step 1 As the bipuser, enter the following command in /home/bipuser:

```
keytool -export -alias tomcat -file tomcat.cer -storepass changeit -keypass changeit
-keystore .keystore
```

Step 2 Switch to the root user:

```
su - root
```

Step 3 Change to the $JAVA_HOME/jre/lib/security directory and enter the following command:

```
keytool -import -alias tomcat -file /home/bipuser/tomcat.cer -keystore cacerts -storepass
changeit
```

Step 4 Confirm that the tomcat entry in /home/bipuser/.keystore is the same entry that is in $JAVA_HOME/jre/lib/security/cacerts. To compare the fingerprints:

a. Execute the following in /home/bipuser:

```
keytool -list -keystore .keystore
Output of keytool -list -keystore .keystore
Keystore type: jks
Keystore provider: SUN
Your keystore contains 1 entry
tomcat, Mar 1, 2007, keyEntry,
Certificate fingerprint (MD5):
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
```

b. Note the Tomcat fingerprint entry, then enter the following in $JAVA_HOME/jre/lib/security:

```
keytool -list -keystore cacerts
Output of keytool -list -keystore cacerts
Keystore type: jks
Keystore provider: SUN
Your keystore contains n entries
entries omitted
tomcat, Mar 1, 2007, trustedCertEntry,
Certificate fingerprint (MD5):
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
entries omitted
```

c. Verify that the Tomcat entry in the cacerts fingerprint is the same as the Tomcat entry in .keystore.

Step 5 Continue with the "Configuring Tomcat" procedure on page 8-14.

Configuring Tomcat

Complete the following steps to configure Tomcat:

Step 1 Switch to the bipuser:

```
su - bipuser
```

Step 2 Change to the following directory:

```
cd $PA_HOME/biplatform/biserver-ce/tomcat/conf/
```

Step 3 Open server.xml with a text editor and add following entry:

```
<!-- Define a SSL Coyote HTTP/1.1 Connector on port 8443 -->
<Connector port="8443" maxThreads="200"
           scheme="https" secure="true" SSLEnabled="true"
           keystoreFile="/home/bipuser/.keystore" keystorePass="changeit"
           clientAuth="false" sslProtocol="TLS"/>
```

If needed, modify port, keystoreFile and keystorePass parameters.

Step 4 Continue with the "Configuring the Jetty Server with the SSL Certificate" procedure on page 8-14.

Configuring the Jetty Server with the SSL Certificate

Complete the following steps to configure the Jetty server with the SSL certificate.

Step 1 As the bipuser, navigate to bipuser home directory:

```
cd $HOME
```

Step 2 Enter the following command:

```
Keytool -keystore .keystore -alias jetty -genkey -keyalg RSA
```

You will be prompted to enter certificate information and passwords to protect the keystore and its keys. The only required information is the fully qualified server host name at the first and last name prompt. For example:

```
Enter keystore password:  changeit
What is your first and last name?
  \[Unknown\]:  localhost
What is the name of your organizational unit?
  \[Unknown\]:
What is the name of your organization?
  \[Unknown\]:
What is the name of your City or Locality?
  \[Unknown\]:
What is the name of your State or Province?
  \[Unknown\]:
What is the two-letter country code for this unit?
  \[Unknown\]:
Is CN=localhost, OU=Unknown, O=Unknown,
L=Unknown, ST=Unknown, C=Unknown correct?
  \[no\]:  yes
Enter key password for <jetty>
        (RETURN if same as keystore password): changeit
```

Note If the keystore already exists in $PA_HOME/biplatform/administration-console/resource/config you can skip the last two steps.

Step 3 After the certificate is created, you might need to locate the file on the disk. This generally is a keystore file in your home directory.

Step 4 Move the keystore file into $PA_HOME/biplatform/administration-console/ resource/config.

This is the default keystore location. If you place the keystore in a different location, be sure to keep this in mind.

Step 5 Continue with the "Configuring the Prime Analytics Properties File" procedure on page 8-15.

Configuring the Prime Analytics Properties File

Complete the following steps to configure the Prime Analytics Properties file:

Step 1 As the bipuser, switch to the following directory:

```
cd $PA_Home/biplatform/administration-console/resource/config
```

Step 2 Open the console.properties file in a text editor and change **console.ssl.enabled** from false to **true**.

By default, the keystore and trustore path is resource/config and password for them is changeit. If you have something different you can edit the SSL section for the correct value. Remember that the default port for the administration console https is 8043, If you want to change it, you can enter the change in the properties file.

Step 3 Switch to the following directory:

```
$PA_Home/biplatform/biserver-ce/tomcat/webapps/pentaho/WEB-INF/classes
```

Step 4 Open the adminprotocol.properties file in a text editor and change the following properties:

- ADMINISTRATION_CONSOLE_PROTOCOL—Change to **https**.
- ADMINISTRATION_CONSOLE_PORT—Enter the same value that was entered in console.ssl.port.number in console.properties.

For example:

```
ADMINISTRATION_CONSOLE_PROTOCOL=https
ADMINISTRATION_CONSOLE_PORT=8143
```

The Prime Analytics Administration Console is now ready to use with SSL.

Step 5 Continue with the "Validating SSL" procedure on page 8-16.

Validating SSL

Complete the following steps to validate SSL with Prime Analytics:

Step 1 Switch to the root user.

```
su - root
```

Step 2 Start the BI platform server:

```
service biplatform start
```

Step 3 Enter following URLs in a browser window:

- User Console: https://<server>:8443
- Administration Console: https://<server>:8443/pentaho/Admin

Step 4 Click **Manage Users and Roles** and verify that the Administration console is displayed.

Customizing Server Parameters

The Prime Analytics database server parameters are stored in a file named postgresql.conf. This file is stored in the root of your database data directory:

```
$PGDATA/postgresql.conf
```

Because TruCQ is built on PostgreSQL, all PostgreSQL configuration parameters are valid. For information about PostgreSQL parameters, see *http://www.postgresql.org/docs/8.3/static/runtime-config.html*

Note The postgresql.conf file has several logging parameters that you might want to adjust, For information, see TruCQ Logging for Problem Resolution, page 8-23.

Enabling and Restricting Network Access

Prime Analytics CQ engine defaults configuration allow anyone on the local machine to access the database, but access from anywhere else is blocked. If you want to restrict access or open it to more systems over the network, see the instructions at:

http://www.postgresql.org/docs/8.3/static/client-authentication.html.

In the default installation, PGDATA is set to /var/opt/primeanalytics/data. Therefore, you must modify $PGDATA/pg_hba.conf and the listen_addresses setting in $PGDATA/postgresql.conf. You might also need to adjust any active system firewalls to allow traffic through them.

TruCQ Server Parameters

Some Prime Analytics server configuration parameters are specific to TruCQ. They control TruCQ behavior. You can modify TruCQ parameters using normal PostgreSQL configuration interfaces:

- Editing the postgresql.conf file
- Using SET command
- Using the pg_settings system view.

Table 8-3 describes the TruCQ general option settings.

Table 8-3 TruCG General Option Settings

Option	Type	Default	Description
enable_cq	boolean	true	Enables CQ functionality in TruCQ. If disabled, TruCQ behaves like standard PostgreSQL: continuous queries are not allowed and stream processing does not occur. Disabling enable_cq allows streams to be created and altered in ways that might not be possible with enable_cq enabled. This can be useful when restoring a database.
cq_block_enqueue	boolean	true	Controls how TruCQ handles internal communication queues. These queues buffer communications between various TruCQ processes. If stream tuples arrive at a very high rate, the buffers might fill up. In the default configuration, TruCQ blocks streams until buffer space is available. If the parameter is disabled, TruCQ will drop data.
cq_cursor_block_enqueue	boolean	false	Controls how TruCQ handles full cursor communication queues. These queues buffer the results returned by cursors on continuous queries. If clients do not promptly fetch results from a cursor, the buffer might fill up. By default, TruCQ drops the newest data tuples when this occurs. This parameter enables the blocking. Enable this parameter when are running performance analyses. If not enabled, your results might be inflated because not all the produced data will be properly consumed.
cq_log_block_enqueue	boolean	false	Enables logging on blocked insertions into internal queues. The log message format is: `enqueue tuple is blocking on tuplechannel = {16349, 0, 1}`

Table 8-3 TruCG General Option Settings (continued)

Option	Type	Default	Description
cq_copy_ignore_errors	boolean	false	Specifies how COPY should handle stream input errors. By default, COPY in PostgreSQL aborts the current transaction when input data errors occur. If this parameter is enabled, TruCQ makes an exception for COPY commands loading data into streams. Erroneous input lines are dropped and logged. This allows the COPY to proceed. Correcting the data to avoid format errors is recommended. However, you can use this parameter as a workaround for erroneous data. If you use this workaround, actively monitor the logs to keep track of how much erroneous data you are receiving, then change your policies as needed.
cq_database	string	cqdb	Specifies the name of the database in which continuous queries are evaluated. You can use only one database for CQ data.
cq_telescope_view_enable	boolean	false	Enables creation of telescoping views.
cq_replication_enqueue_enable	boolean	true	Enables queuing of replicator work in a replicator catalog tables while set.
cq_replicator_enable	boolean	true	Enables the use of the CQ replicator process.
cq_default_serial_enable	boolean	false	Changes CREATE STREAM defaults to WITHOUT PARTIALS.
cq_node_id	integer	0	Unique identifier for a node in a cluster. Should be unique for each cluster node.
cq_replicator_naptime	number	1	Amount of time in seconds between replicator runs.
cq_run_id	number	null	Sets the run ID for this transaction. If used by the user, care should be taken to ensure the uniqueness per transaction.
cq_multi_runtime_auto_dist_enable	boolean	false	If enabled, distributes serial streams on different runtimes. The runtime is determined using the OID of the first with partials predecessor modulo cq_max_runtimes_processes.

Table 8-3 describes the TruCQ resource allocation options.

Table 8-4 TruCQ Resource Allocation Options

Option	Type	Default	Min/Max	Description
cq_buffer_page_size	number	32 KB	8 KB/ 64 KB	Controls the size of a single CQ buffer page. Data tuples are stored in CQ buffer pages in shared memory to reduce the processing required to copy data tuples between TruCQ processes. This also determines the size of each channel segment. (A channel is an internal communication queue.)
cq_max_channel_segments	number	5	1/20	Controls the maximum number of segments allocated to a channel.
cq_max_runtime_processes	integer	1	1/32	Determines the maximum number of CQ runtime processes to start.
cq_max_archiver_processes	integer	1	1/32	Determines the maximum number of CQ archive processes to start.

Table 8-4 TruCQ Resource Allocation Options (continued)

Option	Type	Default	Min/Max	Description
cq_max_archives	integer	32	1/2048	Specifies the number of channels sent to a relation that is expected to run on the system, for example, regular, checkpoint, or correction channels. TruCQ needs this information because each channel is stored in shared memory so work can be run in parallel. The channels are allocated when the database is started. You should over estimate this parameter to avoid having to restart the database if you add more channels.
cq_planner_shmem_size	number	1 MB	N/A	Controls the size of an internal queue used to communicate query plans inside TruCQ. The 1 MB default is appropriate for most applications.
cq_shared_mem_size	number	16 MB	N/A	Controls the amount of memory allocated for internal communication queues in the TruCQ. If you expect significant data bursts in any stream, you might raise this value.

Allocating Database Memory

To prevent individual users from allocating excessive memory that canot be paged out, UNIX limits the amount of shared memory that a process can allocate. By default, these limits are very small. 32 MB is standard. Because of this limitation, the default TruCQ configuration cannot allocate the required shared memory block needed for high-performance. Instead, the defaults are set so that the CQ engine server can start, even with an untuned kernel that has very low shared memory limitations.

Increasing Shared Memory Limits

To start the server after increasing the TruCQ shared memory parameters, such as cq_shared_mem_size, you will likely need to adjust your OS kernel turning to allow that. The following PostgreSQL documentation provides information on how to adjust or remove this limitation by increasing the approximate kernel tunables.

http://www.postgresql.org/docs/8.3/static/kernel-resources.html

You might need to adjust many of the listed parameters if you must support a large setting for the database max_connections beyond the default of 100. On Linux, you can generally change both SHMMAX (in bytes) and SHMALL (in pages, normally 4096 bytes) to be a large portion, but not all, of the system RAM. 50% of the total RAM is normally large enough to hold even a database that is optimized for heavy memory usage, while, at the same time, not too large to prevent swap allocations through shared memory.

Permanent changes to these settings are made by editing /etc/sysctl.conf as the root user. A script that will produce the appropriate settings to allow up to half of the RAM to be used for shared memory is provided below:

```
#!/bin/bash
mem_bytes='awk '/MemTotal:/ { printf "%0.f",$2 * 1024}' /proc/meminfo' mem_max='expr
$mem_bytes / 2' page_size='getconf PAGE_SIZE' shmall='expr $mem_bytes / $page_size' echo
\# Maximum shared segment size in bytes echo kernel.shmmax = $mem_max echo \# Maximum
number of shared memory segments in pages echo kernel.shmall = $shmall
```

The output from this script consists of lines to add to the sysctl.conf file. Here is an example of its output from a system with 8 GB of RAM:

```
# Maximum shared segment size in bytes kernel.shmmax = 4189255680 # Maximum number of
shared memory segments in pages kernel.shmall = 2045535
```

You could redirect the output from the script directly to /etc/sysctl.conf and then run the sysctl program to get the change to take effect. (This example presumes the above was saved as shm.sh.)

```
./shm.sh >> /etc/sysctl.conf
sysctl -p
```

You do not need to reboot, and the settings will be preserved after restarting.

Increasing Shared Memory Allocation

At startup, the server allocates two blocks of memory in which the bulk of the server operations will normally occur. The shared_buffers parameter is used as a database page caching mechanism for regular database access. Using the cq_shared_mem_size option to increasing it will improve performance on regular (non-streaming) queries.

The parameters controlling these two will default to the following (these lines are in two different spots in the file):

```
shared_buffers = 32MB
#cq_shared_mem_size = 16MB
```

The "#" means that this setting is commented out and is not active. To change it, you must remove the # from the beginning of the line, then set a larger value. An increased one would look like this:

```
cq_shared_mem_size = 256MB
```

One way track the settings that are changed from their defaults is to put them at the end of the postgresql.conf file to keep them together, and commenting out (using #) the locations of settings you do not want applied. The primary setting you will normally need to monitor is shared_buffers. This is normally set to some very small value approximately 100 lines into the file, for example:

```
shared_buffers = 32MB # min 128kB or max_connections*16kB
```

Comment it out will avoid any confusion about which setting takes precedence.

```
#shared_buffers = 32MB # min 128kB or max_connections*16kB
```

Note that on 32-bit platforms, these shared memory parameters will be limited to a maximum of 2 GB even if more RAM is available.

Memory Sizing Guidelines

A configuration snippet from a system with 8 GB of RAM is shown below. You could scale the following up or down depending on how the target system compared to that, keeping the same general ratio:

```
shared_buffers = 512MB
effective_cache_size = 4096MB
cq_shared_mem_size = 512MB
checkpoint_segments = 32
```

The proportions here are:

- shared_buffers—1/16 of RAM

- cq_shared_mem_size—1/16 of RAM
- effective_cache_size—1/2 of RAM
- checkpoint_segments—Isn't a memory parameter. It determines the disk allocation for the database write-ahead logs and is critical for good database performance. 32 allocates approximately1.3GB of disk space for them.

You can increase performance by increasing the two shared memory parameters further, with 1/8 of total RAM still being a fairly conservative value for those. The other CQ-related parameters generally do not need to change on larger or smaller systems. If you have a high number of streams or stream-related objects, you might need to increase the cq_max_channel_segments.

Directory Paths

The main Prime Analytics software is installed into /opt/primeanalytics. Never write anything to this area (by default the files are owned by the root user). Reinstalling Prime Analytics will overwrite everything in this directory.

Changing the Database Location

The main directories used by the Prime Analytics are determined by the settings in the /etc/sysconfig/primeanalytics/primea file. This sets a number of environment variables used by other parts of the software:

export PA_HOME=<root directory>/primeanalytics

export ANT_HOME=$PA_HOME/thirdparty/apache-ant-1.7.1

export JAVA_HOME=$PA_HOME/thirdparty/jdk1.6.0_37

export PATH=$JAVA_HOME/bin:$ANT_HOME/bin:$PA_HOME/TruCQ/bin:$PATH

The database and its related configuration files are stored in a directory tree whose location is identified by the PGDATA environment variable. This variable defaults to /var/opt/primeanalytics/msjor/minor/data. The structure is created by the service primeanalytics initdb command.

You can change the PGDATA directory to point to another location as long as you do this before the initdb step. Your database will live in that alternate location.

The only database requirement is that the database storage tree should be a directory named data. The Linux standard of placing data files into /var is followed, but they can be relocated elsewhere in your file system that makes sense.

Using External or Network Storage for the Database

The Prime Analytics database can be stored on any type of storage as long as it obeys the POSIX fsync mechanics. Avoid the NFS and Linux LVM storage types. Use normal ext3 partitions instead

It is also critical that any disk used does not include any write cache that can be lost, which typically means that only battery-backed write caches are acceptable. See http://www.postgresql.org/docs/current/static/wal-reliability.html for more information on this topic.

Disk Space Estimation

The binaries for the Prime Analytics software take approximately 90 MB to install. You will need that much space in /opt/primeanalytics for each version you want to install simultaneously. The usual way to estimate future disk space is to load a useful subset of the desired data type, measure the table size and extrapolate from there.

User Accounts

PGDATA and CQLOG information is only needed for users who stop and start the database. You do not need to worry about them for regular users. User accounts map into the PostgreSQL role structure by default.

Changing the Prime Analytics HTTP Port

By default, users access the Prime Analytics BI platform server through Port 8080. If you want to change it to a different port, you must manually edit the following files. The examples below change the port to 9080.

- tomcat/conf/server.xml

```
<Connector URIEncoding="UTF-8" port="9080" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

- biserver-ce/tomcat/webapps/pentaho/WEB-INF/web.xml

```
<context-param>
             <param-name>fully-qualified-server-url</param-name>
             <param-value>http://testarossa5:9080/pentaho/</param-value>
      </context-param>
```

- biserver-ce/tomcat/webapps/pentaho/jsp/Admin.jsp

```
  <div class="btn_manageusers"><a style="color: #0088c2" title="Manage users and
roles" target="_self"
href=<%="http://"+request.getServerName()+":9080/pentaho/AdministrationConsole"%>>Mana
ge Users and Roles</a></div>
```

Configuring and Managing Logs

The standard way to configure the CQ engine is to edit the postgresql.conf file. You can find detailed PostgreSQL configuration information at:

http://www.postgresql.org/docs/8.3/static/runtime-config.html

PostgreSQL error reporting and logging can be found at:

http://www.postgresql.org/docs/8.3/static/runtime-config-logging.html

TruCQ Logging for Problem Resolution

Additional logging that can be helpful if problems occur include:

- Set log_statement to all. This logs every statement executed in by the CQ engine.
- Set the log_min_messages field to DEBUG.
- Set debug_print_plan to true.
- Set log_connections and log_disconnections to true. This logs all connections and disconnections including the various client programs that connect to the database.
- Turn log_hostname to true. This ensures that connection and disconnection messages include resolved hostnames and not IP addresses, which makes debugging easier. The DNS lookup can cause a performance penalty, particularly if not every host connecting is set up properly in DNS.

These suggestions cause a small performance penalty and create larger log files, particularly if you set log_min_messages to DEBUG. While you should usually avoid increasing the log_min_messages volume, doing so can be valuable if an error is easily reproducible.

Relocating the Prime Analytics Logs

Log files are located in three locations. By default, database logs go to the location indicated by the CQLOG environment (set in /etc/sysconfig/primeanalytics/primea). You can change this by editing the postgresql.conf file.

After you turn the logging_collector on, the database does not send normal errors or messages. It redirects them to the file specified in other postgresql.conf sections through the log_directory and log_filename parameters. Under normal conditions, nothing is printed anywhere else.

However, the init script still outputs to $CQLOG. This can be useful if you have problems when starting the server before the postgresql.conf file is read. The error message only appears in the CQLOG location. This commonly occurs when you try to start the database server while it is running. In this case, it cannot open the file where the rotated logs are located because the first server is using them. It will append output to the cq.log file. $CQLOG normally points to the cq.log and indicates startup failed.

If you wanted all log files to go into a log/ directory under the Prime Analytics user's login, you could initially set this up as follows:

```
$ cd ~primeanalytics
$ mkdir -p logs/db logs/app
```

You would then change the CQLOG setting in /etc/sysconfig/primeanalytics/primea:

```
CQLOG=/var/opt/primeanalytics/logs/db/cqstartup.log
```

You might use a rotated log system for the rest of the logs by adding these lines to the postgresql.conf:

```
log_destination = 'stderr' logging_collector = on log_directory =
'/var/opt/primeanalytics/logs/db' log_filename = 'primeanalytics-%a.log'
log_truncate_on_rotation = on
log_rotation_age = 1440 log_rotation_size = 0
```

You can then create a symbolic link in the application's log directory to point to the new structure:

```
$ cd ~primeanalytics/myapp
$ cd apache-tomcat-5.5.23
$ rm -rf logs $ ln -s ~primeanalytics/logs/app logs
```

TruCQ Log Rotation

In addition to the general logging suggestions provided by the http://www.postgresql.org/docs/8.3/static/runtime-config-logging.html, other log rotation options are discussed at http://www.postgresql.org/docs/8.3/static/logfilemaintenance.html

Here is a subset of a postgresql.conf configuration that creates one log file per day and rotates it every week:

```
#------------------------------------------------------------------------------# ERROR
REPORTING AND LOGGING
#------------------------------------------------------------------------------
# -Where to Log
# log_destination = 'stderr'
# Valid values are combinations of
# stderr, csvlog, syslog and eventlog,
# depending on platform. csvlog
# requires logging_collector to be on.
# This is used when logging to stderr: logging_collector = on
# Enable capturing of stderr and csvlog
# into log files. Required to be on for
# csvlogs. # (change requires restart)
# These are only used if logging_collector is on:
# log_directory = 'pg_log'
# directory where log files are written,
# can be absolute or relative to PGDATA log_filename = 'primeanalytics-%a.log'
# log file name pattern,
# can include strftime() escapes log_truncate_on_rotation = on
# If on, an existing log file of the
# same name as the new log file will be
# truncated rather than appended to.
# But such truncation only occurs on
# time-driven rotation, not on restarts # or size-driven rotation. Default is
# off, meaning append to existing files
# in all cases. log_rotation_age = 1440
# Automatic rotation of logfiles will
# happen after that time. 0 to disable. log_rotation_size = 0
# Automatic rotation of logfiles will
# happen after that much log output.
# 0 to disable.
```

This will write the log files to a pg_log directory under the PGDATA subdirectory with the log_directory setting still at the default:

```
#log_directory = 'pg_log'
```

If you want to separate the database logs from the database, change this with an absolute path name instead.

BI Platform Logs

The BI platform log files are located at $PA_HOME/biplatform/tomcat/logs. These include catalina.out, pentaho.log, and truviso.log. The default setting is to show only errors. However you can perform the same modifications as the other logs by editing:

<*BI Server*>/biserver-ce/tomcat/webapps/pentaho/WEB-INF/classes/log4j.xml

Adjusting TruLink Logging

TruLink logging is performed by the Java log4j system. You can change the number of messages you see by editing the customizations/templates/WEB-INF/classes/log4j.properties file.

The best first step to solving problems in this area is to increase the amount of logging done by the application components. Change the customizations/templates/WEB-INF/classes/log4j.properties file to look like this:

```
log4j.logger.com.primeanalytics=DEBUG, primeanalytics
```

In some scenarios, changing the level to TRACE instead of DEBUG might be useful. Note that TRACE is very verbose and likely to impact performance significantly.

TruLink Log Rotation

Because projects use the log4j system, you can adjust log file size and number using the RollingFileAppender class. See http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/RollingFileAppender.html for complete details.

Here is a sample log4j configuration designed to rotate these logs based on their size:

```
log4j.appender.primeanalytics=org.apache.log4j.RollingFileAppender
log4j.appender.primeanalytics.File=${catalina.home}/logs/primeanalytics.log
log4j.appender.primeanalytics.maxFileSize=1024MB
log4j.appender.primeanalytics.maxBackupIndex=2
log4j.appender.primeanalytics.layout=org.apache.log4j.PatternLayout
log4j.appender.primeanalytics.layout.ConversionPattern=%d [%t] %-5p %c -%m%n
```

Customizing Server Startup Behavior

The Red Hat chkconfig utility allows you to adjust the run levels that start the server automatically. Here is an example of how to add each Prime Analytics service to chkconfig and to make them start automatically on boot up:

```
/sbin/chkconfig --add primeanalytics
/sbin/chkconfig primeanalytics on
/sbin/chkconfig --add trulink /sbin/chkconfig trulink on
```

The services are expected to be active on are run levels 3 and 5.

Handling RPM Conflicts

Most files installed by Prime Analytics RPMs have a path that includes the version number. The files in /opt/primeanalytics are all uniquely named this way. A few configuration files are shared among all versions:

```
/etc/sysconfig/primeanalytics/primea /etc/init.d/primeanalytics /etc/init.d/trulink
/var/opt/primeanalytics/.bash_profile
```

If you modify any of these files, and try to install a new version, your customized version won't be overwritten. Instead, the new version will be created with the suffix ".rpmnew". You will need to manually merge the contents of the two files. For most version upgrades, you can edit /etc/sysconfig/primeanalytics/primea, The diff utility can help you identify what is customized and what has changed in the new version:

```
cd /etc/sysconfig/primeanalytics diff --side-by-side primeanalytics primeanalytics.rpmnew
```

Using multiple versions concurrently

Because of the configuration file conflicts, RPM will reject any attempt to install two versions of the Prime Analytics software at the same time. However, if you are willing to manage those configuration files by hand, you can have more than one version installed. This makes it easier to roll back to an older version if there proves to be a problem with the newer one.

To install the new version and override the conflict warning, use the -force parameter to RPM:

```
rpm -ivh --force primeanalytics*.rpm
```

You can then switch between similar versions mainly by editing primeanalytics_REL setting in /etc/sysconfig/primeanalytics/primea.

You can even have multiple active startup scripts by copying /etc/init.d/primeanalytics to another name, then creating a new entry in the /etc/sysconfig/primeanalytics directory with that same name. Make sure the port numbers and other shared resources are unique so more than one can work at once.

Backing Up and Restoring Prime Analytics

Backing up and restoring Prime Analytics involves three areas:

- User content—This includes dashboards, reports, charts, and analyses.
- Database repository—This includes historical data used to create the user dashboards, reports, and analyses.
- Configuration files

Note All backup and restore procedures apply to the same Prime Analytics version.

User Data

Content created by Prime Analytics users is stored in the file system solutions repository located at:

$PA_HOME/biplatform/biserver-ce/pentaho-solutions

User-created OLAP and reporting models are located in the following directories:

./primeanalytics/biplatform/biserver-ce/pentaho-solutions/admin/resources/metadata
./primeanalytics/biplatform/biserver-ce/pentaho-solutions/system/olap/datasources.xml

Database Repository

The Prime Analytics BI platform contains three databases:

- Hibernate
- Quartz
- Connector metadata

The hibernate database stores BI platform system configuration such as users, user roles, and user settings. The quartz database stores BI platform scheduling information. The connector database stores realtime SQL information used for creating dashboards. The quartz, hibernate, and connector databases are created on TruCQ and are included with the BI platform.

Prime Analytics scripts back up and restore the BI platform and the TruCQ database. These scripts are located in the following directories:

- TruCQ database—$PA_HOME/bin/backup_restore/Database
- BI Platform—$PA_HOME/bin/backup_restore/BIPlatform

The following practices ensure the safety and efficiency of the backup and restore process:

- Stopping the BI platform server before you back up the BI platform is recommended to avoid any transient issues.
- The backup.dir property in backup_restore.properties (available in both the BIPlatform and Database directories) should point to the same shared network directory with write permissions for both the primea and bipuser users.
- Back up the shared network drive periodically and verify a mechanism exists to recover the data.
- Backing up the BI platform and database using CRON jobs during non-peak hours is recommended to minimize the network impact.

The following topics provide details and prerequisites for backing up and restoring Prime Analytics:

- Backing Up the BI Platform, page 8-27
- Backing Up the Database, page 8-28
- Restoring the BI Platform, page 8-28
- Restoring the Database, page 8-29

Backing Up the BI Platform

Before you back up the BI platform, verify that:

- The backup directory is specified in the backup_restore.properties file. that is, backup.dir.
- The bipuser user has write permission to the backup directory specified in backup_restore.properties.

Note The BI platform backup file name is biplatform-TIMESTAMP.tar.gz , where TIMESTAMP is the backup date and time. Do not change the backup file name.

To back up the BI platform:

Step 1 Log into the BI platform as the bipuser.

Step 2 Change to the BI platform backup directory:

```
cd $PA_HOME/bin/backup_restore/BIPlatform
```

Step 3 Start the backup:

```
./backup_biplatform.sh
```

Backing Up the Database

Before you back up the database, verify the following:

- Specify the backup directory in the backup_restore.properties file.
- Verify the user running the database backup script has write permission to the backup directory specified in the backup_restore.properties file.

Note The Prime Analytics database backup does not include TruCQ related configurations, for example, pg_hba.conf, pg_ident.conf, and postgresql.conf located in /var/opt/primea/data/. If a database configuration, such as a TruCQ port, was changed by editing the file, restore will not include the customized port. Database backup also does not back up the application or project directory created by users. You will need to back up these directories using another backup method.

To back up the BI platform:

Step 1 Log into the Prime Analytics TruCQ server as the primea user. If you are switching users, use the su - primea command.

Step 2 Change to the database backup directory:

```
cd $PA_HOME/bin/backup_restore/Database
```

Step 3 Start the backup:

```
./backup_db_repository.sh
```

Restoring the BI Platform

The BI platform restore stops the BI server before performing a restore, so you should advise users the server will not be available during the restore.

To restore the BI platform:

Step 1 Log into the BI platform server as the bipuser.

Step 2 Change to the BI platform restore directory:

```
cd $PA_HOME/bin/backup_restore/BIPlatform
```

Step 3 Start the restore:

```
restore_biplatform.sh <BACKUP_FILE_LOCATION>
```

Where BACKUP_FILE_LOCATION is the tar.gz file with location.

Note Prime Analytics validates the backup file to ensure that it was created by Prime Analytics. If not, the restore will not proceed.

Step 4 After the restore is complete, in the Prime Analytics home page, click the **Refresh** tool above the navigation tree to refresh the GUI display.

Restoring the Database

Before you begin the database restore, verify that the primea user has write permission to the /tmp directory. During the restore, you will stop the BI platform and database servers.

If you are restoring the TruCQ database in an HA configuration where the BIDB (BI platform repository) and TruCQ databases are on different servers, make sure to restore the backup files correctly.

- Restore files starting with bisample-<time>.tar.gz to the TruCQ database
- Restore files starting with repository-<time>.tar.gz to the BIDB database

To restore the Prime Analytics database:

Step 1 Log into the Prime Analytics database (TruCQ) server as the root.

Step 2 Stop the BI platform:

service biplatform stop

Step 3 Stop the TruCQ server:

For TruCQ database:

service trucq stop

For BIDB database (HA):

service bitrucq stop

Step 4 Change to the database restore directory:

```
cd $PA_HOME/bin/backup_restore/Database
```

Step 5 Start the database restore:

./restore_db_repository.sh <BACKUP_FILE_LOCATION>

Where BACKUP_FILE_LOCATION is tar.gz file with location.

Note Prime Analytics validates the backup file to ensure that it was created by Prime Analytics. If not, the restore will not proceed.

Step 6 If you restored the database to a new environment, such as a new hostname or IP address (assuming the BI platform is still the same) manually update the following files with the new values:

- $PA_HOME/biserver-ce /pentaho-solutions/system/applicationContext-spring-security-hibernate.properties
- $PA_HOME/biserver-ce/tomcat/webapps/pentaho/META-INF/context.xml
- $PA_HOME/biserver-ce /pentaho-solutions/system/hibernate/postgresql.hibernate.cfg.xml
- $PA_HOME/biserver-ce /pentaho-solutions/system/quartz/quartz.properties

You might also need to change the entries in the TVDB database because it stores the TruCQ engine details along with server name.

Step 7 After the restore is complete, start the TruCQ database server:

For TruCQ database:

service trucq start

For BIDB database (HA):

service bitrucq start

Step 8 Validate the restore database and delete the original data folder from /tmp.

Step 9 Start the BI platform:

service biplatform start

Starting and Stopping Prime Analytics Servers

Table 8-5 provides commands that you can use to start and stop the Prime Analytics BI platform and TruCQ engine, You must run these commands as the root user.

In general, start and stop the BI platform using the service biplatform command as the root. Alternatively, you can start and stop the BI platform using ./start-primebip.sh and ./stop-primebip.sh as the root.

Table 8-5 Prime Analytics Server Commands

Action	Command
Stop the BI platform.	service biplatform stop
Start the BI platform	service biplatform start
Stop the TruCQ engine	service trucq stop
Start the TruCQ engine	service trucq start

Real Time Chart Parameters

The following topics provide the parameters that appear in real time charts. Prime Analytics real time charts are designed to operate with continuous streams of data. Five real time charts are provided:

Real Time Line Dual Y Axis Report Properties

Table A-1 lists the Real Time Line Dual Y parameters.

Table A-1 Real Time Line DY Chart Properties

Name	Type
Basic Properties	
Chart Caption	String
Chart Sub Caption	String
X Axis Name	String
Primary Y Axis Name	String
Secondary Y Axis Name	String
Data Refresh Interval	Number
Show legend	Boolean (0/1)
Name	String
Path	String
DataAccessId	String
Charts Type	String
Width	String
Height	String
HtmlObject	String

Table A-1 Real Time Line DY Chart Properties (continued)

Name	Type
Advanced Attributes	
Canvas Background Color	Color Code
Canvas Border Thickness	Number
Chart Background Color	Color Code
Chart Border	Boolean (1/0)
Chart Border Color	Color Code
Clear Chart Interval	Number
Max number of data points to plot	Number
Show Y Axis Values	Boolean (0/1)
Y axis decimal places	Number
Secondary Y axis decimal places	Number
X axis show labels	Boolean (0/1)
X axis label display	WRAP, STAGGER, ROTATE, or NONE
X axis slant labels (for ROTATE)	Boolean (0/1)
X axis label step	Number
X axis show values	Boolean (0/1)
X axis rotate values	Boolean (0/1)
Legend Position	RIGHT or BOTTOM
Number format	Boolean (0/1)
Number Prefix	String
Number Suffix	String
Number of Decimal points	Number
Format number scale (K,M)	Boolean (0/1)
Base Font	Font Name
Base Font Size	Number
Base Font Color	Color Code
Chart Link	String
Fusion Chart seriesParam name	String
Fusion Chart categoriesParam name	String
Fusion Chart valueParam name	String
Parameters	String
Listeners	String

Real Time Column Chart Properties

Table A-2 lists the Real Time Column Chart parameters.

Table A-2 Real Time Column Chart Properties

Name	Type
Basic Properties	
Chart Caption	String
Chart Sub Caption	String
X Axis Name	String
Y Axis Name	String
Data Refresh Interval	Number
Show legend	Boolean (0/1)
Name	String
Path	String
DataAccessId	String
Charts Type	String
Width	String
Height	String
HtmlObject	String
Advanced Properties	
Canvas Background Color	Color Code
Canvas Border Thickness	Number (Pixels)
Chart Background Color	Color Code
Chart Border	Boolean (1/0)
Chart Border Color	Color Code
Clear Chart Interval	Number
Max number of data points to plot	Number
Show Y Axis Values	Boolean (0/1)
Y axis decimal places	Number
X axis show labels	Boolean (0/1)
X axis label display	WRAP, STAGGER, ROTATE or NONE
X axis slant labels (for ROTATE)	Boolean (0/1)
X axis label step	Number
X axis show values	Boolean (0/1)
X axis place values inside	Boolean (0/1)
X axis rotate values	Boolean (0/1)
Legend Position	RIGHT or BOTTOM
Number format	Boolean (0/1)

Table A-2 Real Time Column Chart Properties (continued)

Name	Type
Number Prefix	String
Number Suffix	String
Number of Decimal points	Number
Format number scale (K,M)	Boolean (0/1)
Base Font	Font Name
Base Font Size	Number
Base Font Color	Color Code
Chart Link	String
Fusion Chart seriesParam name	String
Fusion Chart categoriesParam name	String
Fusion Chart valueParam name	String
Parameters	String
Listeners	String

Real Time Line Chart Properties

Table A-3 lists the Real Time Line Chart parameters.

Table A-3 Real Time Line Chart Properties

Name	Type
X axis rotate values	Boolean (0/1)
Show legend	Boolean (0/1)
Legend Position	RIGHT or BOTTOM
Number format	Boolean (0/1)
Number Prefix	String
Number Suffix	String
Number of Decimal points	Number
Format number scale (K,M)	Boolean (0/1)
Base Font	Font Name
Base Font Size	Number
Base Font Color	Color Code
Name	String
Path	String
DataAccessId	String
Charts Type	String

Table A-3 Real Time Line Chart Properties (continued)

Name	Type
Chart Link	String
Fusion Chart seriesParam name	String
Fusion Chart categoriesParam name	String
Fusion Chart valueParam name	String
Width	String
Height	String
Parameters	String
HtmlObject	String
Listeners	String

Real Time Stacked Area Properties

Table A-4 lists the Real Time Stacked Area parameters.

Table A-4 Real Time Stacked Area Properties

Name	Type
Canvas Background Color	Color
Canvas Border Thickness	Integer
Chart Background Color	Color
Chart Border	Boolean
Chart Border Color	Color
Stack 100 Percent	Boolean
Show Percent Values	Boolean
Show Percent In Tool Tip	Boolean
Chart Caption	String
Chart Sub Caption	String
X Axis Name	String
Y Axis Name	String
Data Refresh Interval	Integer
Clear Chart Interval	Integer
Max number of data points to plot	Integer
Show Y Axis Values	Boolean
Y axis decimal places	Integer
X axis show labels	Boolean
X axis label display	String
X axis slant labels (for ROTATE)	Boolean

Table A-4 Real Time Stacked Area Properties (continued)

Name	Type
X axis label step	Integer
X axis show values	Boolean
X axis rotate values	Boolean
Show legend	Boolean
Legend Position	String
Number format	Boolean
Number Prefix	String
Number Suffix	String
Number of Decimal points	Integer
Format number scale (K,M)	Boolean
Base Font	String
Base Font Size	Integer
Base Font Color	Color
Name	String
Path	String
DataAccessId	Integer
Charts Type	String
Chart Link	String
Fusion Chart seriesParam name	String
Fusion Chart categoriesParam name	String
Fusion Chart valueParam name	String
Width	Integer
Height	Integer
Parameters	String
HtmlObject	String
Listeners	String

Real Time Angular Gauge Properties

Table A-5 lists the Real Time Angular Gauge parameters.

Table A-5 Angular Guage Properties

Property Name	Type
Chart Background Color	Color Code
Chart Border	Boolean (1/0)
Chart Border Color	Color Code

Table A-5 Angular Guage Properties (continued)

Property Name	Type
Show Gauge Value	Boolean (0/1)
Show Value Below Pivot	Boolean (0/1)
Gauge Lower Limit Label	String
Gauge Upper Limit Label	String
Show Tick Marks	Boolean (0/1)
Show Tick Values	Boolean (0/1)
Show Tick Limits	Boolean (0/1)
Place Tick Inside	Boolean (0/1)
Place Values Inside	Boolean (0/1)
Auto Align Tick Values	Boolean (0/1)
Data Refresh Interval	Number
Number format	Boolean (0/1)
Number Prefix	String
Number Suffix	String
Number of Decimal points	Number
Format number scale (K,M)	Boolean (0/1)
Base Font	Font Name
Base Font Size	Number
Base Font Color	Color Code
Gauge Start Angle	Number (-360-360)
Gauge End Angle	Number (-360-360)
Show Gauge Border	Boolean (0/1)
Gauge Border Color	Color Code or Fill Mix formula (but single token)
Gauge Border Thickness	Number (Pixels)
Name	String
Path	String
DataAccessId	String
Charts Type	String
Chart Link	String
Fusion Chart seriesParam name	String
Fusion Chart categoriesParam name	String
Fusion Chart valueParam name	String
Width	String
Height	String
Parameters	String

Table A-5 Angular Guage Properties (continued)

Property Name	Type
HtmlObject	String
Listeners	String
Range Colors Array	String Array
Intervals Array	String Array

Real Time Chart Parameter Definitions

Table A-6 provides an alphabetical list of parameters used in real time charts.

Table A-6 Real Time Chart Parameters

Parameter	Description
Auto Align Tick Values	Whether to ask the chart to automatically place tick values to reduce overlapping of the tick values with the gauge.
Base Font	Sets the font face (family) of all chart text (data labels, values etc.). If you also specify the outCnvBaseFont attribute, this attribute controls only the font face of text within the chart canvas bounds.
Base Font Color	Sets the base font color, that is, all chart values and names are displayed in the font color provided here.
Base Font Size	Sets the base font size of the chart, that is, all the values and the names in the chart are displayed using the font size provided here.
bgColor	This attribute sets the background color for the chart. You can set any hex color code as the value of this attribute
borderColor	Border color of the chart.
Canvas Background Color	Sets Canvas background color. For a gradient effect, enter the color codes separated by commas.
Canvas Border Thickness	Specifies the canvas border thickness in pixels.
canvasBgColor	Sets Canvas background color. For Gradient effect, enter colors separated by comma.
canvasBorderThickness	Lets you specify canvas border thickness.
caption	Caption of the chart.
Chart Background Color	Sets the chart background color. You can use any hexidecimal color code.
Chart Border	Indicates whether the chart should have a border.
Chart Border Color	Border color of the chart.
Chart Caption	Caption of the chart.
Chart Link	
Chart Sub Caption	Sub-caption of the chart.
ChartCategoriesParam	
ChartLink	
Charts Type	

Table A-6 Real Time Chart Parameters (continued)

Parameter	Description
ChartSeriesParam	
ChartType	
ChartValueParam	
Clear Chart Interval	The time interval when the chart will clear its contents.
Data Refresh Interval	Chart data refresh interval in seconds.
DataAccessId	Data access ID. For real time charts, set this value to 1.
dataAccessID	
decimals	Number of decimal places to which all numbers on the chart will be rounded to. For more details, please see Advanced Charting > Number Formatting section.
Format number scale (K,M)	Adds K (thousands) and M (millions) to a number after truncating and rounding it, for example, if formatNumberScale is set to 1, 1043 becomes 1.04K (with decimals set to two places). The same applies to numbers in millions; an M is added at the end.
formatNumber	This configuration determines whether the numbers displayed on the chart will be formatted using commas, e.g., 40,000 if formatNumber='1' and 40000 ifformatNumber='0'.
formatNumberScale	Configuration whether to add K (thousands) and M (millions) to a number after truncating and rounding it - e.g., if formatNumberScale is set to 1, 1043 will become 1.04K (with decimals set to 2 places). Same with numbers in millions - an M will be added at the end
Fusion Chart categoriesParam name	
Fusion Chart seriesParam name	
Fusion Chart valueParam name	
Gauge Border Color	Color of gauge scale border. You can either specify a hex code here (like 333333), or a single token from gradient fill mix like {dark-20} or {light-30}. The chart will then calculate the border color for each color range individually (based on the color code for each color range).
Gauge Border Thickness	Thickness of gauge scale border.
Gauge End Angle	Angle where the gauge will end drawing. Default value is 0.
Gauge Lower Limit Label	This attribute allows you to display a label instead of the lower limit. For example, in a chart displaying Literacy Rate on a scale of 0-100%, you may need to show the label Low at the starting point of the chart. Upon using this attribute to specify the label, the value 0 will be replaced by Low. Default value: If you do not specify the lowerLimitDisplay attribute, the lower limit value will be shown.
Gauge Start Angle	Angle from where the gauge will start drawing. Default value is 180.
Gauge Upper Limit Label	This attribute allows you to display a label instead of the upper limit. Upon using this attribute, the upper limit of the chart gets replaced by the label specified. Default value: If you do not specify the upperLimitDisplay attribute, the upper limit value will be shown.
Height	Height to the chart
HtmlObject	
Intervals Array	"Array - Intervals to use on the dial. [""0"",""75"",""90"",""100""]. Use for creating XML

Table A-6 Real Time Chart Parameters (continued)

Parameter	Description
labelDisplay	Using this attribute, you can control how your data labels (x-axis labels) will appear on the chart. There are 4 options: WRAP, STAGGER, ROTATE or NONE. WRAP wraps the label text if it's longer than the allotted area. ROTATE rotates the label in vertical or slanted position. STAGGER divides the labels into multiple lines.
labelStep	By default, all the labels are displayed on the chart. However, if you have a set of streaming data (like name of months or days of week), you can opt to show every n-th label for better clarity. This attributes just lets you do so. It allows you to display every n(th) X-axis label.
Legend Position	The legend can be plotted at two positions on the chart - below the chart (BOTTOM) and on the RIGHT side of the chart.
Listeners	
Max number of data points to plot	Specifies the number of data items that you want as history on the chart. For example, when you set it as 50, only 50 data items (per dataset) are displayed on the chart. After the 51st data item arrives, the first data item is deleted and the rest of the data is shifted by one position left. The new data takes the last position.
Name	
Number format	Determines whether the numbers displayed on the chart are formatted using commas, for example, 40,000 if formatNumber=1' and 40000 ifformatNumber=0.
Number of Decimal points	Number of decimal places to which all chart numbers are rounded.
Number Prefix	Adds a prefix to all the numbers visible on the graph. For example, to represent dollar figures, you could set this attribute to ' $' to show 40000 as $40000, for example.
Number Suffix	Adds a suffix to all numbers visible on the graph. For example, to represent all figures quantified as per annum, you could set this attribute to ' /a' to show, for example, 40000/a, 50000/a.
numDisplaySets	This value helps you specify the number of data items that you want as history on the chart. For example, when you set it as 50, only 50 data items (per dataset) will be displayed on the chart. As soon as the 51st data comes in, the first data will be deleted and the rest of the data will be shifted by 1 position left - the new data will come and take the last position.
Parameters	
Path	Chart path.
Place Tick Inside	Whether to place ticks inside the gauge (on the inner radius) or outside?
Place Values Inside	Whether to place tick values inside the gauge?
Primary Y Axis Name	Primary Y-Axis title.
Range Colors Array	"Array - Colors to use in the intervals range. [""FF654F""""", ""F6BD0F"", ""8BBA00""]. Refer to ""CGG Dial Component"" Range Color Array. Value can also be hex color codes.
refreshInterval	For this parameter, you can specify the number of seconds after which the chart will look for new data
rotateValues	If you show data values, you can rotate them using this attribute.
Secondary Y axis decimal places	Decimal places for y-axis values on secondary y-axis.
Secondary Y Axis Name	Secondary Y-Axis title
Show Gauge Border	Whether to show a border around gauge scale?

Table A-6 Real Time Chart Parameters (continued)

Parameter	Description
Show Gauge Value	Whether to show each dial's value?
Show legend	Indicates whether the chart should show a legend. This only applies to multiseries and combination charts.
Show Tick Limits	Whether to show the first and last tick value (i.e., chart lower and upper limit)?
Show Tick Marks	Whether to show tick marks?
Show Tick Values	Whether to show tick values?
Show Value Below Pivot	Whether to show dial value below the pivot or above it?
Show Y Axis Values	The y-axis of charts is divided into vertical sections using div (divisional) lines. Each div line assumes a value based on its position. Using this attribute you can specify whether to show the div line (y-axis) values.
showBorder	Whether to show a border around the chart or not?
showLabels	It sets the configuration whether the x-axis labels will be displayed or not.
showLegend	Whether to show legend for the chart (only multi-series and combination charts)?
showPercentInToolTip	Whether to show percentage figures in tooltip.
showPercentValues	Whether to show percentage figures in data values of the chart.
showValues	Sets the configuration whether data values will be displayed along with the data plot on chart.
slantLabels	If you show rotated labels on a chart, this attribute lets you set the configuration whether the labels will show as slanted labels or fully vertical ones.
stack100Percent	"Setting this attribute to 1 helps in depicting the values in percentage figures. When this attribute is set to 1 the chart considers the total of the stacked dataplot as 100% and the Y-Axis shows a scale from 0% to 100%."
subCaption	Sub-caption of the chart.
Width	Chart width.
X axis label display	Controls how data labels (x-axis labels) appear in the chart. WRAP wraps the label text if it is longer than the allotted area. ROTATE rotates the label in vertical or slanted position. STAGGER divides the labels into multiple lines.
X axis label step	By default, all the labels are displayed on the chart. However, if you have a set of streaming data (such as month names or days of the week), you can opt to show every n-th label for better clarity using this property. The property allows you to display every n(th) X-axis label.
X Axis Name	X-Axis Title of the Chart.
X axis place values inside	If you've opted to show data values, you can show them inside the columns using this attribute. By default, the data values show outside the column.
X axis rotate values	If you show data values, you can rotate them using this attribute.
X axis show labels	Specifies whether the x-axis labels are displayed.
X axis show values	Indicates whether data values are displayed along with the data plot on chart.
X axis slant labels (for ROTATE)	If you selected rotated labels on chart, this attribute determines whether the labels appear as slanted vertical labels.
xAxisName	X-Axis Title of the Chart.

Table A-6 Real Time Chart Parameters (continued)

Parameter	Description
Y axis decimal places	Decimal places for y-axis values.
Y Axis Name	Y-Axis Title of the chart.
yAxisValueDecimals	Decimal places for y-axis values.

Sample Applications

Prime Analytics includes two sample applications showing examples of network data displayed using the Prime Analytics BI platform dashboard and analytics components. Although the applications are fairly simple, they provide examples to give you an understanding of the general concepts and mechanics that go into dashboard creation, and allow you to begin building applications for your network.

Sample applications are described in the following topics:

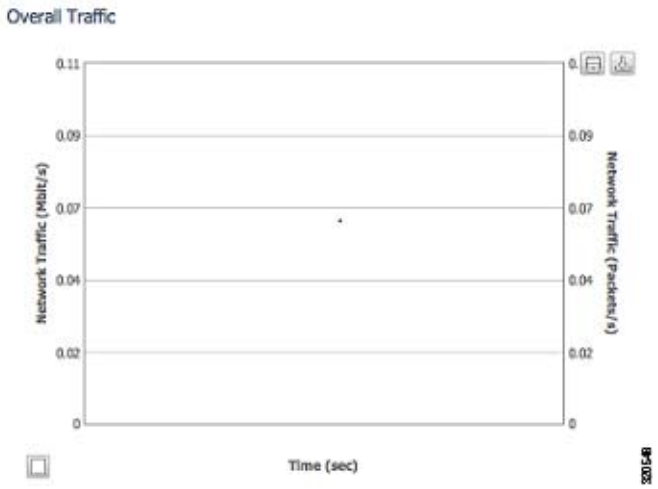
NetFlow Sample Application

The Network Summary sample dashboard shows network traffic based on a continuous NetFlow query stream. The dashboard runs off a local log file to simulate live traffic.

Components

The dashboard includes the following components:

- Overall Traffic—Displays the overall network traffic in the Real Time Dual Y chart component. One Y axis displays overall traffic in megabits per second. The other Y axis displays the overall traffic in packets per second. The data is refreshed every second.



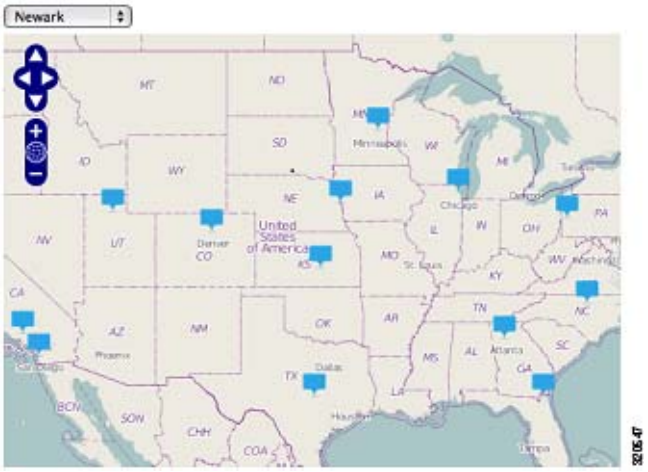
- Top 10 Traffic by IP—Displays the source and destination hosts with the highest traffic in Mb/s. The data is displayed in a three-column table. The data is refreshed every 10 seconds.

Top 10 Traffic by IP

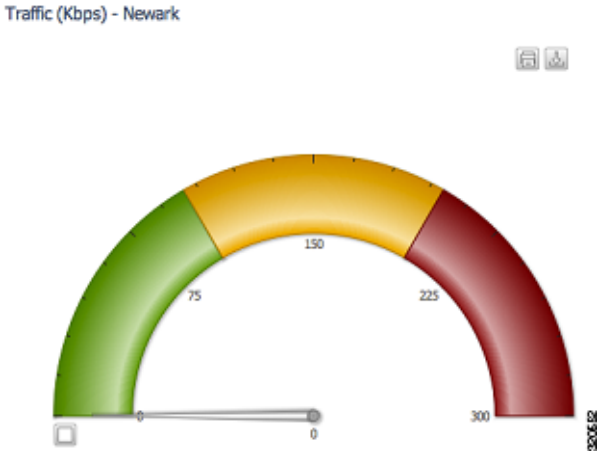
Source IP	Destination IP	Traffic Mbps
60.1.6.10	50.1.6.145	4.32
60.1.0.10	50.1.0.111	4.21
60.1.4.10	50.1.4.153	3.89
60.1.4.10	50.1.4.147	3.79
60.1.2.10	50.1.2.136	3.6
60.1.5.10	50.1.5.111	3.54
60.1.1.10	50.1.1.109	3.38
60.1.0.10	50.1.0.121	3.37
60.1.1.10	50.1.1.157	3.3
60.1.3.10	50.1.3.152	3.09

Showing 1 to 10 of 10 entries

- City Traffic—Displays traffic for individual cities. Dashboard components used to display city traffic include:
  - City Selector—Allows users to choose a city using a map from OpenStreetMap. (See http://www.openstreetmap.org for information.) Using a US map, users can select cities using the city drop down field or choosing a city on the map. (You can implement Google Maps using the GoogleMapAPI. Refer to the Google website documentation for information.)



  - Traffic - [*City*]—Displays traffic in kilobits per second using the Real Time Angular chart.



– Traffic - [*City*]—Displays traffic in kilobits per second using the Real Time Line chart.

– ToS Distribution - [*City*]—Displays a Type of Service distribution using the CCC Pie chart.

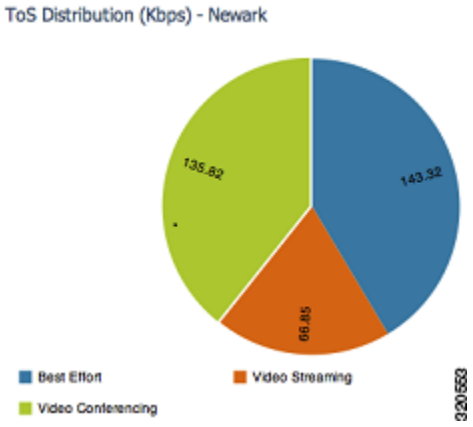


Table B-1 lists the components that comprise the NetFlow Summary sample application.

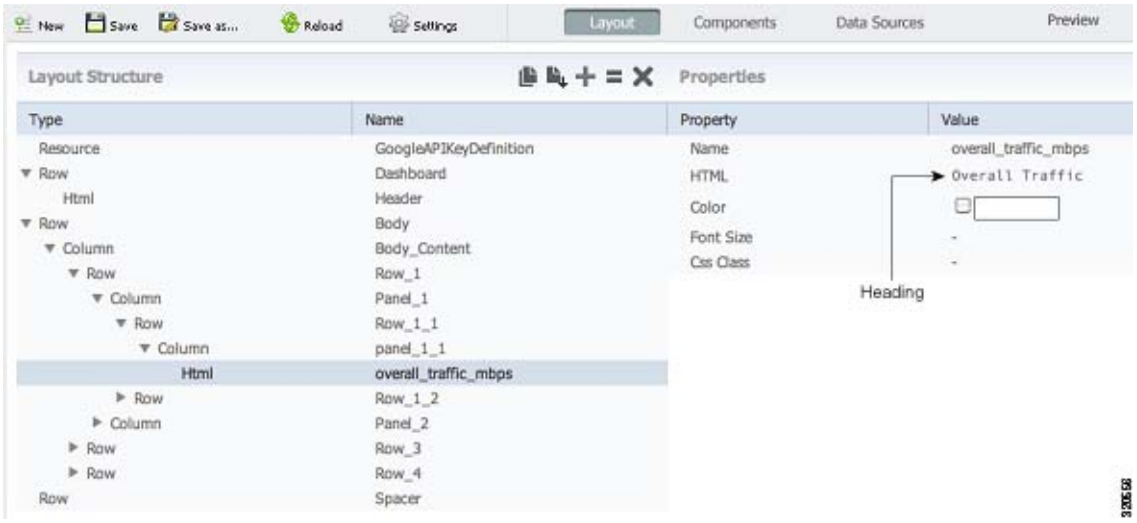
Table B-1 NetFlow Summary Dashboard Components

Dashboard Component	Component Name	Type	Group
General Traffic	overall_traffic	Real Time Line Dual Y	Real Time Charts
Top 10 Traffic by IP	top_10_traffic_table	Table Component	Others
Map	map_header	Text Component	Others
	traffic_map	Map Component	Custom
	param_city_name	Parameter	Generic
	select_city	Select Component	Selects
Traffic - [*city*]	traffic_gauge_header	Text Component	Real Time Charts
	city_traffic_gauge_kbps	Real Time Angular Gauge	Real Time Charts
Traffic - [*city*]	traffic_column_chart_header	Text Component	Others
	city_traffic_kpbs	Real Time Column	Real Time Charts
Tos Distribution	tos_pie_header	Text Component	Others
	tos_by_city	CCC Pie Chart	Charts

Layout

The layout, shown below, is based on a 4x4 template with an additional row added. When creating layouts for your dashboards, a good approach is to choose a template that is closest to your needs, and then customizing it.

To provide uniform labeling placement, labels are placed in separate rows.In the example below, An HTML element with the text, Overall Traffic, is placed in the Row_1_1 row and panel_1_1 column, If you expand the second column: Panel_2 > Row_1_3, Panel_2_1, you will see an HTML component with the second column heading text, Top 10 Traffic by IP. If desired, you can link the HTML element to a Cascading Style Sheet so all text is displayed uniformly.



Data Source

The data sources for the sample NetFlow application are shown below. The data source captured in the screen below is only for static content:

- top_10_traffic_query
  - Top 10 Traffic by IP Table
  - CityDropdownData: City dropdown
  - CityData: Map Component
  - ToSData: ToS Distribution (Kbps) - City Pie Chart

The data source for real-time charts are selected on the Components tab. Choose **Edit Components > Real Time Charts > Real Time Line Dual Y (overall_traffic)**. Under Path Property, click **View SQL** to see a list of available continuous queries. At the bottom click ^ to select the continuous query that you want to use for the chart.

NOTE: You might have to take a few screenshots for selecting the Continuous Query datasource Data sources are defined outside the BI platform. In the sample application, SQL Queries list the data files used to simulate continuous query data. When you set up actual continuous query data sources, you will not set it up under the User Console Data Sources workspace. For information, see Setting Up Continuous Query Data Sources, page 4-8.

Changing Open Street to Google Maps

The following steps show you how to replace the existing Open street map component in the sample NetFlow application with Google maps. Before you begin, you need to get a Google Maps API key. The key is available from Google. See the Google website for details.

Step 1 Open the Network Summary dashboard in Edit mode:

a. Under Browse, click **NetFlow**.

b. Under Files, click **Network Summary**.

c. Click the **Edit** tool.

Step 2 Click the **Layout** tab.

Step 3 Click the **Add Resource** tool (+).

Step 4 Select the Resource Type, **JavaScript**, and the value **Code Snippet**.

Step 5 Click **OK**.

Step 6 In the Name field, enter **GoogleAPIKeyDefinition.**

Step 7 In the Resource Code field, click the button to the right.

Step 8 In the Edit window, enter your Google maps API key as follows:

var API_KEY = 'YOUR_ KEY'; (*Your corporate/personal Google API map key provided by Google*).

Step 9 Click **OK**.

Step 10 Click the **Save** tool to save the changes.

Step 11 Click the **Components** tab.

Step 12 Select the Group, Custom, then choose the Map Component, usa_map.

Step 13 Change the value in property Map Engine from open to **google**.

Step 14 Click **Save**.

Step 15 Validate the change by open the dashboard

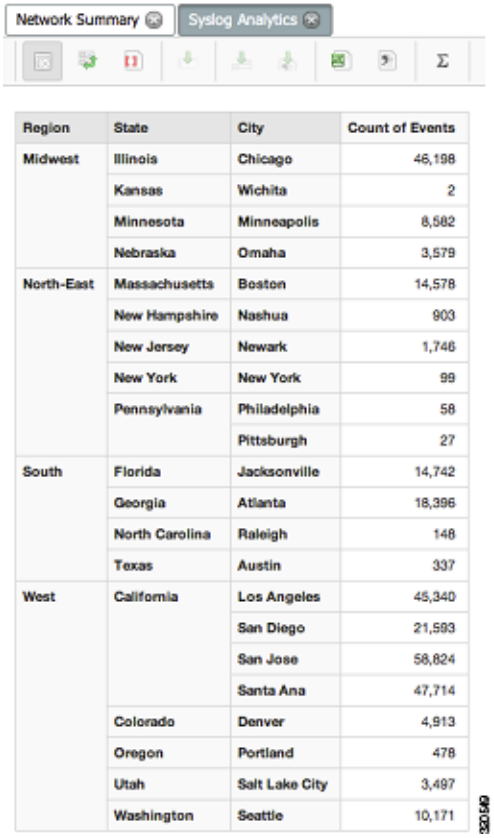
Syslog Analytics Sample Application

The sample syslog application shows how the Prime Analytics BI platform analytics can be used to display network data. Prime Analytics Analytics is based on the Pentaho Mondrian online analytical processing (OLAP) data model. OLAP allows you to drill into and cross-tabulate information in many different ways and from multiple perspectives.

By default, the sample syslog displays the Region, State, and City dimensions with the Count of Events measure. To add additional dates and priorities, select the Syslog Analytics file and choose **Edit**. In the Editing: Syslog Analytics file, add the additional dimensions:

- Date—All, Year, Quarter, Month
- Priority—All, Priority Category, Priority Name

Each added dimension causes a recalculation to occur and results are displayed in the analytics table.



Region	State	City	Count of Events
Midwest	Illinois	Chicago	46,198
	Kansas	Wichita	2
	Minnesota	Minneapolis	8,582
	Nebraska	Omaha	3,579
North-East	Massachusetts	Boston	14,578
	New Hampshire	Nashua	903
	New Jersey	Newark	1,746
	New York	New York	99
	Pennsylvania	Philadelphia	58
		Pittsburgh	27
South	Florida	Jacksonville	14,742
	Georgia	Atlanta	18,396
	North Carolina	Raleigh	148
	Texas	Austin	337
West	California	Los Angeles	45,340
		San Diego	21,593
		San Jose	58,824
		Santa Ana	47,714
	Colorado	Denver	4,913
	Oregon	Portland	478
	Utah	Salt Lake City	3,497
	Washington	Seattle	10,171

320549

Additionally, you can drill down and display the individual syslog data records by clicking **Drill Through on Cell** or **Drill Through on Cell on to CSV** on the Analytics toolbar, then clicking the table. In the dialog, choose the dimensions you want to see, then click **OK**. A second table displays the details of every syslog record for the dimensions you selected.

Region	State	Count of Events
Midwest	Illinois	RP/0/RP1/CPU0:Aug 6 01:19:47.204 EST: spm[371]: %SECURITY-SPM-3-ERR_MSG_GEN : nvram not availab
Midwest	Illinois	*Aug 5 22:31:32.286: %SYS-1-CPUFALLINGTHRESHOLD: Threshold: Total CPU Utilization(Total/Intr) 70%/0
Midwest	Illinois	*Aug 5 23:58:41.802: %SYS-1-CPUFALLINGTHRESHOLD: Threshold: Total CPU Utilization(Total/Intr) 69%/0
Midwest	Illinois	RP/0/RP1/CPU0:Aug 6 06:10:16.068 EST: nvram[75]: %MEDIA-NVRAM-3-CORRUPT : Corrupt nvram. Format with "erase nvram: forma
Midwest	Illinois	DRP/0/4/CPU0:Aug 6 08:38:42.079 : exec[65711]: %SECURITY-login-6-AUTHEN_SUCCESS : Successfully authenticated user 'mbirkner' from '192.168.239.26' on 'vty
Midwest	Illinois	RP/0/RP1/CPU0:Aug 6 18:11:37.590 EST: sysmgr[91]: %OS-SYSMGR-3-ERROR : spm_server(1) (jid 371) exited, will be respawned with a delay (slow-resta
Midwest	Illinois	RP/0/RP1/CPU0:Aug 6 22:02:19.783 EST: nvram[75]: %MEDIA-NVRAM-3-CORRUPT : Corrupt nvram. Format with "erase nvram: forma

320555

The strength of the OLAP analytics model is the ability to consolidate or expand and drill down data in many different ways. While the Syslog Analytics sample application includes only a few dimensions and one measure, it should demonstrate the capability of analytics to show network data in a variety of ways.

Syslog and NetFlow Live Sample Application

The Syslog and NetFlow Live Dashboard sample application allows you to view simulated live NetFlow and syslog data, then connect the sample to your NAM and NGA devices and generate live syslog and NetFlow traffic to the sample dashboard.

The NetFlow Syslog Live Dashboard sample is located in the Samples/NetFlow Syslog Live directory in the User Console directory. The sample provides the following data:

- Cisco NetFlow Generator Appliance (NGA)—NetFlow Sum of Bytes and Sum of Packets by Time.
- Cisco Network Analysis Module (NAM)—Syslog Count of Events by Time.
- NGA and NAM
  - NGA NetFlow—Sum of Bytes
  - NAM syslog—Actual Value by Time

The data is displayed using a dashboard comprised of the Real Time Line Dual Y and Real Time Column chart components. (For information about creating and editing dashboards, see Chapter 5, “Creating Dashboards.”)

The NetFlow and syslog data streams are installed on your TruCQ engine. The schema can be viewed in the following location:

$PA_HOME/biplatform/sampledatagenerator/netflow_namsyslog/customizations/db/ nfsyslog.sql

The Syslog and NetFlow Live Dashboard connectors are installed in:

$PA_HOME/biplatform/sampledatagenerator/netflow_namsyslog

You can change these data sources as needed to ensure the fields that are consumed match your input.

Configuring the Syslog and NetFlow Live Dashboard for Live Data

Complete the following steps to connect the Syslog and NetFlow Live Dashboard to NAM and NGA devices generating syslog and NetFlow traffic on your network:

Step 1 Update the NAM_SYSLOG PORT and NGA_NETFLOW PORT in include-handlers.xml, located at:

`$PA_HOME/biplatform/sampledatagenerator/netflow_namsyslog/customizations/templates/include-handlers.xml.`

Port 514 is the default NAM syslog port; Port 3000 is the default NGA NetFlow port.

Step 2 Update the include-handlers.xml for the NetFlow fields that are consumed:

`$PA_HOME/biplatform/sampledatagenerator/netflow_namsyslog/customizations/templates/include-handlers.xml`

Step 3 Update the syslog_parsing_rules.xml for the syslog fields that are consumed:

`$PA_HOME/biplatform/sampledatagenerator/netflow_namsyslog/syslog_parsing_rules.xml)`

Step 4 Log into your NAM device and direct syslog traffic to the NAM_SYSLOG PORT configured in Step 1.

Step 5 Log into your NGA device and direct NetFlow traffic to the NGA_NETFLOW PORT configured in Step 1.

Step 6 As the root user, navigate to the following directory:

`$PA_HOME`/biplatform/sampledatagenerator/netflow_namsyslog

Step 7 Load the environment variables:

```
$PA_HOME/bin/pa_env.sh
```

Step 8 Run the build:

```
$PA_HOME/bin/build-local.sh
```

Note If you receive errors, the most likely cause is the environment variables are not set. Repeat Step 7 to load all the variables.

Step 9 Start the build:

```
$PA_HOME/bin/start.sh
```

The connectors begin listening on the ports configured in Step 1.

Note If you receive an address already in use error, the port configured in Step 1 is already in use. Either choose a new port or stop the service that is using that port.

Note To see if data is flowing into the port, you can use the tcpdump port <*port number*> command, for example, **tcpdump port 514**.

Step 10 Log into the Prime Analytics BI platform (see Logging Into the User Console, page 3-1).

Step 11 Display the Syslog and NetFlow Live Dashboard and observe your live syslog and NetFlow traffic.

Advanced Topics

This section contains advanced Prime Analytics topics, including

Customizing the Chart Data Set Interpretation

Dashboards include many visual elements. Of these, charts are of the most important because they present your data visually and allow your data to be interpreted quickly and easily. Charts have two Boolean parameters that might be difficult to understand at first: Crosstab mode and Series in rows. Both parameters are set to false by default.

If Crosstab mode is set to true, the chart expects a matrix dataset where series and categories create the matrix rows and columns and the data populates the matrix. The first data column is used for either series or category labels. If Series in rows is set to true, a series is a matrix row and the categories are set in columns. The following tables show how the data is interpreted using the Crosstab mode and Series in rows parameters. In the tables:

- S = series
- C = categories (X axis label for a vertical graph)
- ... = direction the table can be extended
- Shaded cells = column headers

In this example:

- Crosstab mode = true
- Series in rows = true

	C1	C2	...
S1	D	D	...
S2	D	D	...
S3	D	D	...
...	...	...	...

In this example:

- Crosstab mode = true
- Series in rows = false

	S1	S2	...
C1	D	D	...
C2	D	D	...
C3	D	D	...
...	...	...	...

Notice that the column headers will show in your chart when Crosstab mode is true. If Crosstab mode is false, the chart expects a three column dataset where each row corresponds to exactly one data point. In this case, the Series in rows parameter determines the column order. If Series in rows is false, the expected order is Series, Category, Data. If Series in rows is true, the expected order is Category, Series, Data. The following tables show how the data is interpreted when Crosstab mode is false.

Note In the CCC charts, use text strings or floating points for category labels. If you use integers for category labels, the charts might become corrupted.

Note In general, MDX queries and SQL queries with a GROUP BY clause are used to generate data sets in cross tab mode. Other SQL queries are more likely to generate data sets that aren't in cross tab mode.

Note Internally the CCC library uses the values 0,1, … n to refer to the categories along the X axis. If you also use these numbers as a category label, the data point will be drawn in the wrong category. If you need numerical values along the category axis, use floating-point formatted numbers.

If crosstab mode is false, the column headers (names) are not part of the dataset and therefore are not visible in the chart. In this example:

- Crosstab mode = false
- Series in rows = false

S	C	D
S	C	D
S	C	D
...	...	...

In this example:

- Crosstab mode = false
- Series in rows = true

C	S	D
C	S	D
C	S	D
...	...	...

Customizing CCC Charts Using Extension Points

The Pentaho Community Chart Components provide an extensive list of options under Advanced Properties. These options include extension points that allow you to set the font family, font size, and text orientation for different chart text components of the chart. When you select an extension point, you can create a list of name value pairs. The name has the format A_B where A is the item you want to customize and B is the property you want to customizes. For example, if you want to customize the font of title A = titleLabel and the property B = font, the parameter to set is titleLabel_font.

Common item types and the properties you can set as an indented list include:

- Text items, for example, titleLabel, xAxisLabel, yAxisLabel, legend, barLabel, pieLabel:
  - font—For example, 13 px serif.
  - fillStyle—For example, yellow.
  - textAngle—An angle in radians (you can to turn the horizontal axis labels at an angle to prevent them from overlapping with one another).

    Note If you change the textAngle of the xAxisLabel, you probably also need to set the textAlignment to Left to prevent the text from crossing the xAxis.

  - textAlign—For example, left, center, right.
  - textBaseline—For example, top, bottom.
  - textStyle—For example, black.
  - text—A function used to modify or format the text, for example, function(d) {return d.substr(5)} to limit the length of the text to the first five characters.
- dot (the marker used in a dot chart):
  - fillStyle—For example, white.
  - shape—For example, square.
  - shapeRadius—For example, 4
- line-items (line, xAxis, yAxis, dot, xAxisRule, yAxisRule):
  - lineWidth—For example, 0.5.
  - fillStyle—For example, green or #00ff00.
  - strokeStyle—For example, blue.
- bar (the marker used in a bar chart)
- pie (for the pie-charts):
  - innerRadius—For example, 10.
  - strokeStyle—For example, white.

Adding Interactivity with JavaScript functions

You can insert JavaScript functions in many chart locations. If you want to insert a function use the syntax:

```
function() { <body> return result; }
```

Do not put a = in front of the function and do not put a ';' after the closing bracket, otherwise the function will be interpreted as a text string instead of a function. If you want to insert a complex function in the dashboard, put the function in a separate resource file (.js). The JavaScript that you enter will be included in your dashboard file without thorough validation. If the JavaScript has a scoping error, for example, too many or too few closing brackets, your dashboard file might become corrupted and lead to unpredictable results or errors. If you put the JavaScript in a separate .js file, the file will be discarded if it contains a scoping error, and other components of your dashboard will not be affected.

Detailed descriptions of most of these components and their parameters can be found in the Pentaho solution repository. In the user console, go to the folder: BI Developer Examples/CDF/Documentation, or from the File menu, choose Open and navigate to the BI Developer Examples directory.

Note If the BI developer Examples does not show in your repository, edit the index.xml file in this folder and set visibility to true (or ask your administrator to do so).

Visibility of Updates in Continuous Queries

A continuous query (CQ) typically runs for a long and indefinite amount of time. An important question that arises for such long-running CQs is how and when they see updates to tables. For example, consider a CQ that involves a join between stream S and table T. Suppose that table T is updated from time to time on no particular schedule. The question is when each update is visible to the CQ; essentially how often should the system take a fresh snapshot of the committed table set. In theory, two extremes are possible:

- Never—Each CQ does not see any updates to participating tables during its lifetime.
- Immediately—Each CQ sees all updates instantly.

The first option is not desirable because a CQ can live forever and the updates can represent important information. However, the second option has two problems. First, the CQ can involve a join with a subquery over the table being updated. Recomputing every subquery for every incoming stream tuple impacts performance. Recomputation (refreshing a cache) is not a problem for most common situations where the CQ simply performs an index lookup. Updating the snapshot visibility is easy to do. However, a second and more important problem with immediate snapshots is that they can cause non-deterministic and non-repeatable results where a given chunk of data in a window can see multiple versions of the same record.

The solution is to make updates to tables visible in a streaming query only when windows advance and are not seen between window advances. For most queries, the application developer does not need to do anything to enable the update visibility. For some queries, however, where a complex subquery exists that caches computations over a table, the cache is only recomputed when the special built-in cq_cache_inval() function is invoked. It is up to the application developer to ensure this, and the typical pattern is to deploy on a table that can be modified, a trigger that calls the cq_cache_inval() function. On update of the table, the trigger is fired, and that results in the system recomputing the cache on the next window boundary.