Informatica

Informatica® Intelligent Streaming
10.1.1

Big Data Streaming User Guide

Informatica Intelligent Streaming Big Data Streaming User Guide
10.1.1
December 2016

See patents at https://www.informatica.com/legal/patents.html.

DISCLAIMER: Informatica LLC provides this documentation "as is" without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of noninfringement, merchantability, or use for a particular purpose. Informatica LLC does not warrant that this software or documentation is error free. The information provided in this software or documentation may include technical inaccuracies or typographical errors. The information in this software and documentation is subject to change at any time without notice.

NOTICES

This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.
2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, please report them to us in writing at Informatica LLC 2100 Seaport Blvd. Redwood City, CA 94063.

INFORMATICA LLC PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2018-11-09

Table of Contents

##

##

Preface

The Informatica Intelligent Streaming User Guide provides information about how to configure and run streaming mappings on a Spark engine in a Hadoop environment.

Informatica Resources

Informatica Network

Informatica Network hosts Informatica Global Customer Support, the Informatica Knowledge Base, and other product resources. To access Informatica Network, visit https://network.informatica.com.

As a member, you can:

- Access all of your Informatica resources in one place.
- Search the Knowledge Base for product resources, including documentation, FAQs, and best practices.
- View product availability information.
- Review your support cases.
- Find your local Informatica User Group Network and collaborate with your peers.

Informatica Knowledge Base

Use the Informatica Knowledge Base to search Informatica Network for product resources such as documentation, how-to articles, best practices, and PAMs.

To access the Knowledge Base, visit https://kb.informatica.com. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at KB_Feedback@informatica.com.

Informatica Documentation

To get the latest documentation for your product, browse the Informatica Knowledge Base at https://kb.informatica.com/_layouts/ProductDocumentation/Page/ProductDocumentSearch.aspx.

If you have questions, comments, or ideas about this documentation, contact the Informatica Documentation team through email at infa_documentation@informatica.com.

Informatica Product Availability Matrixes

Product Availability Matrixes (PAMs) indicate the versions of operating systems, databases, and other types of data sources and targets that a product release supports. If you are an Informatica Network member, you can access PAMs at https://network.informatica.com/community/informatica-network/product-availability-matrices.

Informatica Velocity

Informatica Velocity is a collection of tips and best practices developed by Informatica Professional Services. Developed from the real-world experience of hundreds of data management projects, Informatica Velocity represents the collective knowledge of our consultants who have worked with organizations from around the world to plan, develop, deploy, and maintain successful data management solutions.

If you are an Informatica Network member, you can access Informatica Velocity resources at http://velocity.informatica.com.

If you have questions, comments, or ideas about Informatica Velocity, contact Informatica Professional Services at ips@informatica.com.

Informatica Marketplace

The Informatica Marketplace is a forum where you can find solutions that augment, extend, or enhance your Informatica implementations. By leveraging any of the hundreds of solutions from Informatica developers and partners, you can improve your productivity and speed up time to implementation on your projects. You can access Informatica Marketplace at https://marketplace.informatica.com.

Informatica Global Customer Support

You can contact a Global Support Center by telephone or through Online Support on Informatica Network.

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
http://www.informatica.com/us/services-and-training/support-services/global-support-centers.

If you are an Informatica Network member, you can use Online Support at http://network.informatica.com.

CHAPTER 1

Introduction to Intelligent Streaming

This chapter includes the following topics:

Intelligent Streaming Overview

You can subscribe to sources that stream data and process data as it becomes available. Streaming sources stream data as messages. Use Informatica Intelligent Streaming mappings to collect the streamed data, build the business logic for the data, and push the logic to a Spark engine for processing. The Spark engine uses Spark Streaming to process data. The Spark engine reads the data, divides the data into micro batches and publishes it.

You can create streaming mappings to stream machine, device, and social media data. You can stream data from sources such as JMS providers and Apache Kafka brokers. Use a Kafka connection or JMS connection to access the data as it becomes available.

You can stream the following types of data:

- Application and infrastructure log data
- Change data(CDC) from databases
- Clickstreams from web servers
- Geo-spatial data from devices
- Sensor data
- Time series data
- Supervisory Control And Data Acquisition (SCADA) data
- Message bus data
- Programmable logic controller (PLC) data
- Point of sale data from devices

You can stream data to different types of targets, such as Kafka, HDFS, NoSQL databases and enterprise messaging systems.

Intelligent Streaming is built on the Informatica Big Data Platform platform and extends the platform to provide streaming capabilities. Intelligent Streaming uses Spark Streaming to process streamed data. It uses YARN to manage the resources on a Spark cluster more efficiently and uses third-parties distributions to connect to and push job processing to a Hadoop environment.

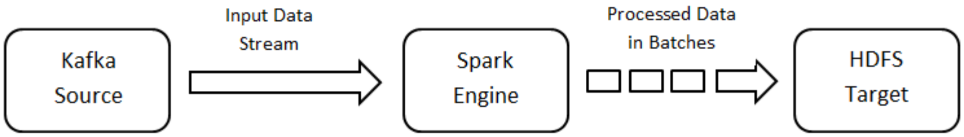
Use Informatica Developer (the Developer tool) to create streaming mappings. Use the Hadoop run-time environment and the Spark engine to run the mapping. You can configure high availability to run the streaming mappings on the Hadoop cluster.

For more information about the Spark engine, see the *Informatica Big Data Management User Guide*.

Streaming Process

A streaming mapping receives data from unbounded data sources. An unbounded data source is one where data is continuously flowing in and there is no definite boundary. Sources stream data as events. The Spark engine receives the input data streams and divides the data into micro batches. The Spark engine processes the data and publishes data in batches.

The following image shows how the Spark engine receives data and publishes data in batches:



The Spark engine uses Spark Streaming to process data that it receives in batches. Spark Streaming receives data from streaming sources such as Kafka and divides the data into discretized streams or DStreams. DStreams are a series of continuous streams of Resilient Distributed Datasets (RDD).

For more information about Spark Streaming, see the Apache Spark documentation.

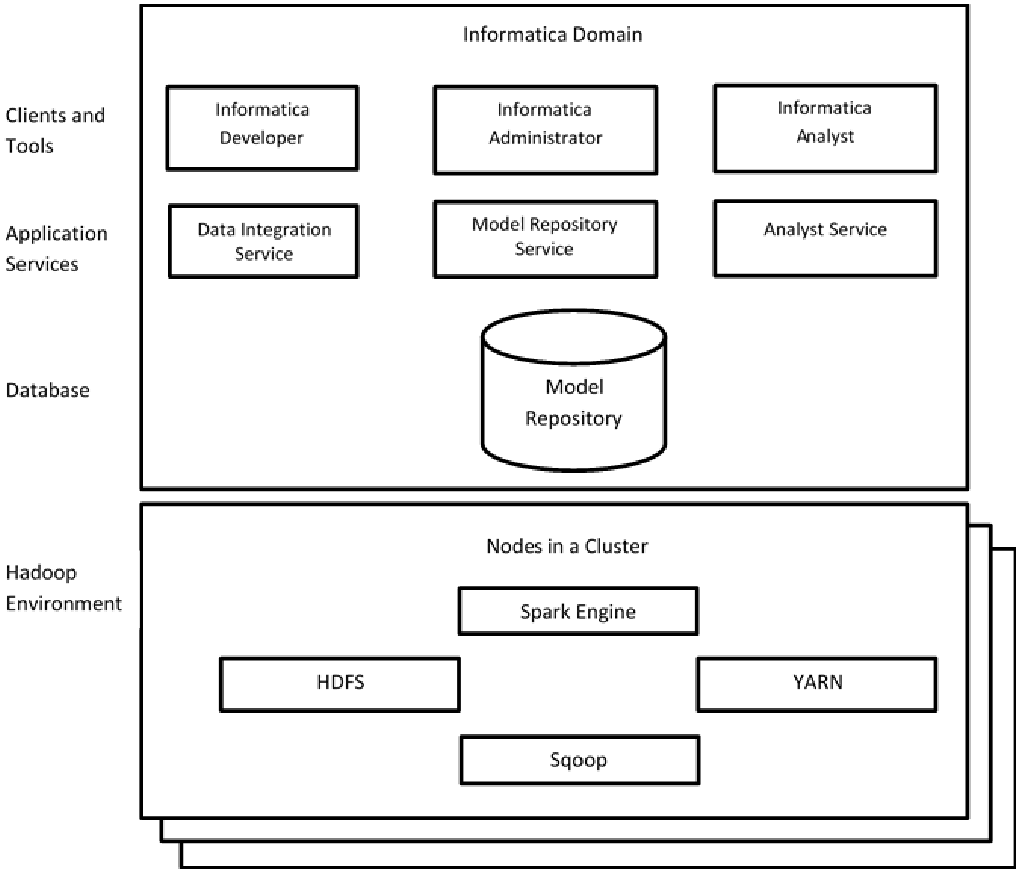
You can perform the following high-level tasks in a streaming mapping:

1. Identify sources from which you need to stream data. You can access data that is in XML, JSON, or Avro format.
   You can use Kafka and JMS sources to connect to multiple big data sources.
2. Configure the mapping and mapping logic to transform the data.
3. Run the mapping on the Spark engine in the Hadoop environment.
4. Write the data to Kafka targets, HDFS complex files, HBase, and Hive tables.
5. Monitor the status of your processing jobs. You can view monitoring statistics for your processing jobs in the Monitoring tool.

Component Architecture

The Intelligent Streaming components for a streaming mapping include client tools, application services, repositories, and third-party tools.

The following image shows the components that Intelligent Streaming uses for Spark streaming mappings:



Clients and Tools

Based on your product license, you can use multiple Informatica tools and clients to manage streaming mappings.

Use the following tools to manage streaming mappings:

Informatica Administrator

Monitor the status of mappings on the Monitoring tab of the Administrator tool. The Monitoring tab of the Administrator tool is called the Monitoring tool.

Informatica Developer

Create and run mappings on the Spark engine from the Developer tool.

Informatica Analyst

Create rules in Informatica Analyst and run the rules as mapplets in a streaming mapping.

Application Services

Intelligent Streaming uses application services in the Informatica domain to process data. The application services depend on the task you perform.

Intelligent Streaming uses the following application services when you create and run streaming mappings:

Data Integration Service

The Data Integration Service processes mappings on the Spark engine in the Hadoop environment. The Data Integration Service retrieves metadata from the Model repository when you run a Developer tool mapping. The Developer tool connects to the Data Integration Service to run mappings.

Model Repository Service

The Model Repository Service manages the Model repository. The Model Repository Service connects to the Model repository when you run a mapping.

Analyst Service

The Analyst Service runs the Analyst tool in the Informatica domain. The Analyst Service manages the connections between service components and the users that have access to the Analyst tool.

Repository

Intelligent Streaming includes a repository to store data related to connections and source metadata. Intelligent Streaming uses application services in the Informatica domain to access data in the repository.

Intelligent Streaming stores Spark streaming mappings in the Model repository. You can manage the Model repository in the Developer tool.

Third-Party Applications

Intelligent Streaming uses third-parties distributions to connect to a Spark engine on a Hadoop cluster.

Intelligent Streaming pushes job processing to the Spark engine. It uses YARN to manage the resources on a Spark cluster more efficiently.

Example

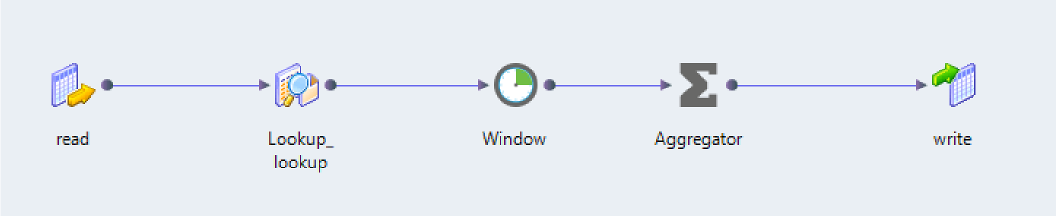
You run the IT department of a major bank that has millions of customers. You want to monitor network activity in real time. You need to collect network activity data from various sources such as firewalls or network devices to improve security and prevent attacks. The network activity data includes Denial of Service (DoS) attacks and failed login attempts made by customers. The network activity data is written to Kafka queues.

You perform the following tasks:

1. Create a streaming mapping to read data from the Kafka queues that stream data in JSON, XML, or Avro formats.
2. Configure the mapping. Add a Lookup transformation to get data from a particular customer ID. Add a Window transformation to accumulate the streamed data into data groups before processing the data.
3. Process the data. Perform aggregations on the data from the customer ID.

4. Monitor jobs. Monitor statistics for the mapping job on the Monitoring tab of the Administrator tool.

The following image shows the mapping:



CHAPTER 2

Intelligent Streaming Configuration

This chapter includes the following topics:

Intelligent Streaming Configuration Overview

Informatica Intelligent Streaming is installed with Informatica Big Data Management. You enable Intelligent Streaming with a license key.

After you install Big Data Management, perform the post-installation tasks to ensure that Big Data Management runs properly. Also perform additional configuration tasks for Spark runtime engine to run mappings and for Hadoop distribution the cluster uses.

Before you create and use JMS connections and JMS data objects in Streaming mappings, complete the required prerequisites.

For more information about the post-installation tasks and JDBC driver JAR files for Sqoop connectivity, see the *Informatica Big Data Management Installation and Configuration Guide*.

Prerequisites to Create a JMS Connection and a JMS Data Object

Before you create a JMS connection or data object, you must include the JMS provider client libraries on the machine running Informatica Intelligent Streaming.

To create a JMS connections, you require the following JAR files from the IBM MQ server:

- com.ibm.mq.allclient.jar
- com.ibm.mq.axis2.jar
- com.ibm.mq.commonservices.jar

- com.ibm.mq.defaultconfig.jar
- com.ibm.mq.headers.jar
- com.ibm.mq.jar
- com.ibm.mq.jmqi.jar
- com.ibm.mq.jms.Nojndi.jar
- com.ibm.mq.pcf.jar
- com.ibm.mq.pcf.jar
- fscontext.jar
- jms.jar
- providerutil.jar
- com.ibm.mq.postcard.jar
- com.ibm.mq.soap.jar
- com.ibm.mq.tools.ras.jar
- com.ibm.mq.traceControl.jar
- com.ibm.mqjms.jar
- jndi.jar
- jta.jar
- ldap.jar
- rmm.jar

Place the client JAR files in the following location:

- Developer tool installation directory: `<Developer Tool Installation Directory>/clients/DeveloperClient/connectors/thirdparty/infa.jms/common`

Prerequisites to Use a JMS Connection and a JMS Data Object

Before you use a JMS connection and JMS data object in your streaming mapping, perform the following tasks:

1. Place the client JAR files in all the data nodes and namenodes of the Hadoop cluster where Intelligent Streaming is installed.

   For example, place the JAR files in the `<Installation Directory>/opt/Informatica/services/shared/hadoop/<hadoop distribution type>/lib` directory.
2. Make sure that the IBM MQ client is installed on all the cluster nodes.
3. In the `hadoop.Env.properties` file edit the `infaspark.executor.thirdparty.jars` and the `infaspark.driver.cluster.mode.thirdparty.jars` Hadoop environment properties to include the location of the JAR files you copied from the IBM MQ server.

   For example, edit the properties to include the following location:

   ```
   $HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.allclient.jar,$HADOOP_NODE_HADOOP_DIST/lib/
   com.ibm.mq.axis2.jar,
   ```

```
$HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.commonservices.jar,
$HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.defaultconfig.jar,
$HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.headers.jar,$HADOOP_NODE_HADOOP_DIST/lib/
com.ibm.mq.jar,
$HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.jmqi.jar,$HADOOP_NODE_HADOOP_DIST/lib/
com.ibm.mq.jms.Nojndi.jar,
$HADOOP_NODE_HADOOP_DIST/lib/com.ibm.mq.pcf.jar,$HADOOP_NODE_HADOOP_DIST/lib/
com.ibm.mq.pcf.jar,
$HADOOP_NODE_HADOOP_DIST/lib/fscontext.jar,$HADOOP_NODE_HADOOP_DIST/lib/jms.jar,
$HADOOP_NODE_HADOOP_DIST/lib/providerutil.jar
```

The `hadoop.Env.properties` file is located in the `<Informatica Installation Directory>/source/services/shared/hadoop/<hadoop distribution type>/infaConf` directory.

CHAPTER 3

Connections

This chapter includes the following topics:

Connections Overview

Define the connections that you want to use to access data in Kafka brokers, JMS servers, HDFS sequence files, Hive tables, or HBase resources. You can create the connections using the Developer tool and infacmd.

You can create the following types of connections:

Hadoop

Create a Hadoop connection to run mappings on the Hadoop cluster. Select the Hadoop connection if you select the Hadoop run-time environment. You must also select the Hadoop connection to validate a mapping to run on the Hadoop cluster.

For more information about the Hadoop connection properties, see the *Informatica Big Data Management User Guide*.

HBase

Create an HBase connection to write data to an HBase resource.

HDFS

Create an HDFS connection to write data to an HDFS sequence file.

Hive

Create a Hive connection to write data to Hive tables.

JDBC

Create a JDBC connection when you perform a lookup on a relational database using Sqoop.

For more information about the JDBC connection properties, see the *Informatica Big Data Management User Guide*.

Messaging

Create a Messaging connection to access data as it becomes available, and to run a streaming mapping on a Spark engine. You can create the following types of messaging connections:

- Kafka. Create a Kafka connection to read from or write to a Kafka broker.
- JMS. Create a JMS connection to read from or write to a JMS server.

HBase Connection

Create an HBase connection to write data to an HBase table.

You can create and manage an HBase connection in the Developer tool or through infacmd.

General Properties

The following table describes the general connection properties for the HBase connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /
ID	The string that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	The description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Select the domain name.
Type	The connection type. Select HBase.

Connection Properties

The following table describes the connection properties for the HBase connection:

Property	Description
ZooKeeper Host(s)	Name of the machine that hosts the ZooKeeper server. When the ZooKeeper runs in the replicated mode, specify a comma-separated list of servers in the ZooKeeper quorum servers. If the TCP connection to the server breaks, the client connects to a different server in the quorum.
ZooKeeper Port	Port number of the machine that hosts the ZooKeeper server.

Creating an HBASE Connection using Informatica Command Line Program

You can use the infacmd command line program to create an HBASE connection. Access the command from the `<Informatica installation directory>/clients/DeveloperClient/infacmd` directory.

On UNIX, run the following command:

```
sh infacmd.sh ISP createConnection
```

On Windows, run the following command:

```
infacmd.bat ISP createConnection
```

Enter connection options in the following format:

```
... -o option_name='value' option_name='value' ...
```

For example, On UNIX you can run the following command to create an HBASE connection on a cluster where Kerberos is not enabled:

```
sh infacmd.sh createConnection -dn Domain -un Administrator -pd Administrator -cn <connection
name> -cid <connection ID> -ct HBase -o ZOOKEEPERHOSTS=<host name> ZOOKEEPERPORT=<port>
ISKERBEROSENABLED=false
```

Run the following command to create an HBASE connection on a cluster where Kerberos is enabled:

```
sh infacmd.sh createConnection -dn Domain -un Administrator -pd Administrator -cn <connection
name> -cid <connection ID> -ct HBase -o ZOOKEEPERHOSTS=<host name> ZOOKEEPERPORT=<port>
ISKERBEROSENABLED=true hbaseMasterPrincipal=hbase/<domain.name>@<YOUR-REALM>
hbaseRegionServerPrincipal=hbase_rs/<domain.name>@<YOUR-REALM>
```

For more information about the CreateConnection command and the HBASE connection options, see the *Informatica Command Reference Guide*.

HDFS Connection

Create an HDFS connection to write data to an HDFS sequence file.

You can create and manage an HDFS connection in the Developer tool or through infacmd.

General Properties

The following table describes the general connection properties for the HDFS connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /`
ID	The string that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.

Property	Description
Description	The description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Select the domain name.
Type	The connection type. Select Hadoop File System.

Connection Properties

The following table describes the connection properties for the HDFS connection:

Property	Description
User Name	User name to access HDFS.
NameNode URI	The URI to access HDFS. Use the following format to specify the NameNode URI in Cloudera and Hortonworks distributions: `hdfs://<namenode>:<port>` Where - `<namenode>` is the host name or IP address of the NameNode. - `<port>` is the port that the NameNode listens for remote procedure calls (RPC). Use one of the following formats to specify the NameNode URI in MapR distribution: - `maprfs:///` - `maprfs:///mapr/my.cluster.com/` Where `my.cluster.com` is the cluster name that you specify in the `mapr-clusters.conf` file.

Creating an HDFS Connection using Informatica Command Line Program

You can use the `infacmd` command line program to create an HDFS connection. Access the command from the `<Informatica installation directory>/clients/DeveloperClient/infacmd` directory.

On UNIX, run the following command:

```
sh infacmd.sh ISP createConnection
```

On Windows, run the following command:

```
infacmd.bat ISP createConnection
```

Enter connection options in the following format:

```
... -o option_name='value' option_name='value' ...
```

For example, run the following command to create an HDFS connection on UNIX:

```
sh infacmd.sh ISP createConnection -dn testDomain -un Administrator -pd Administrator -cn
HDFS_CDH -ct HadoopFileSystem -o nameNodeURL= hdfs://<namenode>:<port> userName=root
```

For more information about the CreateConnection command and HDFS connection options, see the *Informatica Command Reference Guide*.

Hive Connection

Create an Hive connection to write data to a Hive table.

You can create and manage a Hive connection in the Developer tool or through infacmd.

General Properties

The following table describes the general connection properties for the Hive connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /
ID	The string that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	The description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Hive.

Hiveserver Connection Properties

The following table describes the connection properties that you configure to access Hive as a target:

Property	Description
Metadata Connection String	The JDBC connection URI used to access the metadata from the Hadoop server. You can use PowerExchange® for Hive to communicate with a HiveServer service or HiveServer2 service. To connect to HiveServer, specify the connection string in the following format: `jdbc:hive2://<hostname>:<port>/<db>` Where - <hostname> is name or IP address of the machine on which HiveServer2 runs. - <port> is the port number on which HiveServer2 listens. - <db> is the database name to which you want to connect. If you do not provide the database name, the Data Integration Service uses the default database details. To connect to HiveServer 2, use the connection string format that Apache Hive implements for that specific Hadoop Distribution. For more information about Apache Hive connection string formats, see the Apache Hive documentation.
Bypass Hive JDBC Server	JDBC driver mode. Select the check box to use the embedded JDBC driver mode. To use the JDBC embedded mode, perform the following tasks: - Verify that Hive client and Informatica services are installed on the same machine. - Configure the Hive connection properties to run mappings in the Hadoop cluster. If you choose the non-embedded mode, you must configure the Data Access Connection String. Informatica recommends that you use the JDBC embedded mode.
Data Access Connection String	The connection string to access data from the Hadoop data store. To connect to HiveServer, specify the non-embedded JDBC mode connection string in the following format: `jdbc:hive2://<hostname>:<port>/<db>` Where - <hostname> is name or IP address of the machine on which HiveServer2 runs. - <port> is the port number on which HiveServer2 listens. - <db> is the database to which you want to connect. If you do not provide the database name, the Data Integration Service uses the default database details. To connect to HiveServer 2, use the connection string format that Apache Hive implements for the specific Hadoop Distribution. For more information about Apache Hive connection string formats, see the Apache Hive documentation.

HiveCLI Connection Properties

The following table describes the Hive connection properties that you configure to use the Hive connection to run streaming mappings:

Property	Description
Database Name	Namespace for tables. Use the name `default` for tables that do not have a specified database name.
Default FS URI	The URI to access the default Hadoop Distributed File System. Use the following connection URI: `hdfs://<node name>:<port>` Where - <node name> is the host name or IP address of the NameNode. - <port> is the port on which the NameNode listens for remote procedure calls (RPC). If the Hadoop cluster runs MapR, use the following URI to access the MapR File system: `maprfs:///`
JobTracker/Yarn Resource Manager URI	The service within Hadoop that submits the MapReduce tasks to specific nodes in the cluster. Use the following format: `<hostname>:<port>` Where - <hostname> is the host name or IP address of the JobTracker or Yarn resource manager. - <port> is the port on which the JobTracker or YARN resource manager listens for remote procedure calls (RPC). If the cluster uses MapR with YARN, use the value specified in the `yarn.resourcemanager.address` property in yarn-site.xml. You can find `yarn-site.xml` in the following directory on the NameNode of the cluster: `/opt/mapr/hadoop/hadoop-2.5.1/etc/hadoop` MapR with MapReduce 1 supports a highly available JobTracker. If you are using MapR distribution, define the JobTracker URI in the following format: `maprfs:///`
Hive Warehouse Directory on HDFS	The absolute HDFS file path of the default database for the warehouse that is local to the cluster. For example, the following file path specifies a local warehouse: `/user/hive/warehouse` For Cloudera CDH, if the Metastore Execution Mode is remote, then the file path must match the file path specified by the Hive Metastore Service on the Hadoop cluster. For MapR, use the value specified for the `hive.metastore.warehouse.dir` property in `hive-site.xml`. You can find `hive-site.xml` in the following directory on the node that runs HiveServer2: `/opt/mapr/hive/hive-0.13/conf`

Property	Description

Advanced Hive/Hadoop Properties	Configures or overrides Hive or Hadoop cluster properties in hive-site.xml on the machine on which the Data Integration Service runs. You can specify multiple properties.
	Select **Edit** to specify the name and value for the property. The property appears in the following format:
	`<property1>=<value>`
	Where
	- <property1> is a Hive or Hadoop property in hive-site.xml.
	- <value> is the value of the Hive or Hadoop property.
	When you specify multiple properties, `&:` appears as the property separator.
	The maximum length for the format is 1 MB.
	If you enter a required property for a Hive connection, it overrides the property that you configure in the Advanced Hive/Hadoop Properties.
	The Data Integration Service adds or sets these properties for each map-reduce job. You can verify these properties in the JobConf of each mapper and reducer job. Access the JobConf of each job from the Jobtracker URL under each map-reduce job.
	The Data Integration Service writes messages for these properties to the Data Integration Service logs. The Data Integration Service must have the log tracing level set to log each row or have the log tracing level set to verbose initialization tracing.
	For example, specify the following properties to control and limit the number of reducers to run a mapping job:
	`mapred.reduce.tasks=2&:hive.exec.reducers.max=10`

Temporary Table Compression Codec	Hadoop compression library for a compression codec class name.
Codec Class Name	Codec class name that enables data compression and improves performance on temporary staging tables.
Metastore Execution Mode	Controls whether to connect to a remote metastore or a local metastore. By default, local is selected. For a local metastore, you must specify the Metastore Database URI, Driver, Username, and Password. For a remote metastore, you must specify only the Remote Metastore URI.

Metastore Database URI	The JDBC connection URI used to access the data store in a local metastore setup. Use the following connection URI:
	`jdbc:<datastore type>://<node name>:<port>/<database name>`
	where
	- <node name> is the host name or IP address of the data store.
	- <data store type> is the type of the data store.
	- <port> is the port on which the data store listens for remote procedure calls (RPC).
	- <database name> is the name of the database.
	For example, the following URI specifies a local metastore that uses MySQL as a data store:
	`jdbc:mysql://hostname23:3306/metastore`
	For MapR, use the value specified for the `javax.jdo.option.ConnectionURL` property in `hive-site.xml`. You can find hive-site.xml in the following directory on the node where HiveServer 2 runs: `/opt/mapr/hive/hive-0.13/conf`

Property	Description
Metastore Database Driver	Driver class name for the JDBC data store. For example, the following class name specifies a MySQL driver: `com.mysql.jdbc.Driver` For MapR, use the value specified for the `javax.jdo.option.ConnectionDriverName` property in `hive-site.xml`. You can find `hive-site.xml` in the following directory on the node where HiveServer 2 runs: `/opt/mapr/hive/hive-0.13/conf`
Metastore Database Username	The metastore database user name. For MapR, use the value specified for the `javax.jdo.option.ConnectionUserName` property in `hive-site.xml`. You can find `hive-site.xml` in the following directory on the node where HiveServer 2 runs: `/opt/mapr/hive/hive-0.13/conf`
Metastore Database Password	The password for the metastore user name. For MapR, use the value specified for the `javax.jdo.option.ConnectionPassword` property in `hive-site.xml`. You can find `hive-site.xml` in the following directory on the node where HiveServer 2 runs: `/opt/mapr/hive/hive-0.13/conf`
Remote Metastore URI	The metastore URI used to access metadata in a remote metastore setup. For a remote metastore, you must specify Thrift server details. Use the following connection URI: `thrift://<hostname>:<port>` Where - <hostname> is name or IP address of the Thrift metastore server. - <port> is the port on which Thrift server is listening. For MapR, use the value specified for the `hive.metastore.uris` property in `hive-site.xml`. You can find `hive-site.xml` in the following directory on the node where HiveServer 2 runs: `/opt/mapr/hive/hive-0.13/conf`

Creating a Hive Connection using Informatica Command Line Program

You can use the `infacmd` command line program to create a Hive connection. Access the command from the `<Informatica installation directory>/clients/DeveloperClient/infacmd` directory.

On UNIX, run the following command:

```
sh infacmd.sh ISP createConnection
```

On Windows, run the following command:

```
infacmd.bat ISP createConnection
```

Enter connection options in the following format:

```
... -o option_name='value' option_name='value' ...
```

For example, run the following command to create a Hive connection on UNIX:

```
sh infacmd.sh ISP createConnection -dn testDomain -un Administrator -pd Administrator -cn
<connection name> -ct HIVE -o relationalSourceAndTarget=true metadataConnString=j
jdbc:hive://<hostname>:<port>/<db> bypassHiveJDBCServer=false connectString= jdbc:hive://
<hostname>:<port>/<db> databaseName=default defaultFSURI= hdfs://<node name>:<port>
```

```
jobTrackerURI= <jobtrackername>:<port> hiveWarehouseDirectoryOnHDFS=/user/hive/warehouse
METASTOREEXECUTIONMODE=remote ENABLEQUOTES=false username=cehdp password=cehdp
remoteMetastoreURI= thrift://<hostname>:<port> PUSHDOWNMODE=true sqlAuthorized=false'
```

For more information about the CreateConnection command and Hive connection options, see the *Informatica Command Reference Guide*.

Kafka Connection

The Kafka connection is a Messaging connection. Use the Kafka connection to access an Apache Kafka broker as a source or a target. You can create and manage a Kafka connection in the Developer tool or through infacmd.

The Kafka broker maintains configuration information in Apache ZooKeeper. Apache ZooKeeper is a centralized service that maintains the configuration information of the Apache brokers.

When you configure a Kafka connection, you configure the following properties:

- The list of Kafka brokers that the connection reads from or writes to.
- The list of ZooKeeper hosts that maintain the configuration.
- The number of seconds the Integration Service attempts to reconnect to the database if the connection fails.

General Properties

The following table describes the general connection properties for the Kafka connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: ~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /
ID	The string that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	The description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Select the domain name.
Type	The connection type. Select Messaging/Kafka.

Kafka Broker Properties

The following table describes the Kafka broker properties for the Kafka connection:

Property	Description
Kafka Broker List	The IP address and port combinations of the Kafka messaging system broker list. The IP address and port combination has the following format: `<IP Address>:<port>` You can enter multiple comma-separated IP address and port combinations.
ZooKeeper Host Port List	The IP address and port combination of Apache ZooKeeper which maintains the configuration of the Kafka messaging broker. The IP address and port combination has the following format: `<IP Address>:<port>` You can enter multiple comma-separated IP address and port combinations.
Retry Timeout	Number of seconds the Integration Service attempts to reconnect to the Kafka broker to read data. If the source or target is not available for the time you specify, the mapping execution stops to avoid any data loss.

Note: When you click **Test Connection** to verify that you entered the connection properties correctly, the Data Integration Service verifies the connection to ZooKeeper and does not verify the connection to the Kafka broker.

Creating a Kafka Connection using Informatica Command Line Program

You can use the `infacmd` command line program to create a Kafka connection. Access the command from the `<Informatica installation directory>/clients/DeveloperClient/infacmd` directory.

On UNIX, run the following command:

```
sh infacmd.sh ISP createConnection
```

On Windows, run the following command:

```
infacmd.bat ISP createConnection
```

Enter connection options in the following format:

```
... -o option_name='value' option_name='value' ...
```

For example, run the following command to create a Kafka connection on UNIX:

```
sh infacmd.sh ISP createConnection -dn testDomain -un Administrator -pd Administrator -cn
Kafka_CDH -ct Kafka -o zkHostPortList=<host1:port1>,<host2:port2>,<host3:port3>
kfkBrkList=<host1:port1>,<host2:port2>,<host3:port3>
```

For more information about the CreateConnection command, see the *Informatica Command Reference Guide*.

JMS Connection

The JMS connection is a Messaging connection. Use the JMS connection to read messages from a JMS server. You can create and manage a JMS connection in the Developer tool or through infacmd.

General Properties

The following table describes the general connection properties for the JMS connection:

Property	Description
Name	The name of the connection. The name is not case sensitive and must be unique within the domain. You can change this property after you create the connection. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /`
ID	The string that the Data Integration Service uses to identify the connection. The ID is not case sensitive. It must be 255 characters or less and must be unique in the domain. You cannot change this property after you create the connection. Default value is the connection name.
Description	The description of the connection. Enter a string that you can use to identify the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection. Select the domain name.
Type	The connection type. Select Messaging/JMS.

Connection Properties

The following table describes the connection properties for the JMS connection:

Property	Description
Connection URL	The location and port of the JMS provider on which to connect. For example: `tcp://jndiserverA:61616`
User Name	User name to the connection factory.
Password	The password of the user account that you use to connect to the connection factory.
JNDI Context Factory	The JMS provider specific initial JNDI context factory implementation for connecting to the JNDI service. This value is a fully qualified class name of the Initial Context Factory. For example, the class name of the Initial Context Factory for ActiveMQ is `org.apache.activemq.jndi.ActiveMQInitialContextFactory` For more information, see the documentation of the JMS provider.

Property	Description
JNDI Package Prefixes	A colon-delimited list of package prefixes to use when loading URL context factories. These are the package prefixes for the name of the factory class that will create a URL context factory. For more information about the values, see the documentation of the JMS provider.
JMS Connection Factory	The name of the object in the JNDI server that enables the JMS Client to create JMS connections. For example, `jms/QCF` or `jmsSalesSystem`.

Creating a JMS Connection using Informatica Command Line Program

You can use the `infacmd` command line program to create a JMS connection. Access the command from the `<Informatica installation directory>/clients/DeveloperClient/infacmd` directory.

On UNIX, run the following command:

```
sh infacmd.sh ISP createConnection
```

On Windows, run the following command:

```
infacmd.bat ISP createConnection
```

Enter connection options in the following format:

```
... -o option_name='value' option_name='value' ...
```

For example, run the following command to create a JMS connection on UNIX:

```
createConnection -dn Domain -un Administrator -pd Administrator -cn jmsconn -ct Jms -o
url='<url_val> username=<u_val> password=<pass_val> contextFactory='<cf_val>
packagePrefixes=<pp_val> jmsConnectionFactory=<jcf_val>
```

For more information about the CreateConnection command, see the *Informatica Command Reference Guide*.

CHAPTER 4

Sources and Targets in a Streaming Mapping

This chapter includes the following topics:

Sources and Targets in a Streaming Mapping Overview

You can access log file data, sensor data, Supervisory Control And Data Acquisition (SCADA) data, message bus data, Programmable logic controller (PLC) data on the Spark engine in the Hadoop environment.

You can create physical data objects to access the different types of data. Based on the type of source you are reading from or target that you are writing to, you can create the following data objects:

Complex file data object

A representation of a file in the Hadoop file system. Create a complex file data object to write data to an HDFS sequence file.

For more information about complex file data objects, see the *Informatica PowerExchange for HDFS User Guide*.

HBase data object

A physical data object that represents data in an HBase resource. Create an HBase data object to connect to an HBase data target.

JMS data object

A physical data object that accesses a JMS server. You can create a JMS data object to read from or write to a JMS server.

Kafka data object

A physical data object that accesses a Kafka broker. You can create a Kafka data object to read from or write to a Kafka broker.

Relational data object

A physical data object that you can use to access a relational table. You can create a relational object to connect to a Hive data target.

For more information about relational data objects, see the *Informatica Developer Tool Guide*.

Complex File Data Objects

Create a complex file data object with an HDFS connection to write data to an HDFS sequence file.

When you create a complex file data object, a read and write operation is created. To use the complex file data object as a target in streaming mappings, configure the complex file data object write operation properties. You can select the mapping environment and run the mappings on the Spark engine of the Hadoop environment.

When you configure the data operation properties, specify the file format in which the complex file data object writes data to the HDFS sequence file. You can specify XML, JSON, or Avro as format. When you specify XML format, you must provide an XSD file. When you specify JSON or Avro format, you must provide a sample file.

Complex File Data Object Overview Properties

The Data Integration Service uses overview properties when it reads data from or writes data to a complex file.

Overview properties include general properties that apply to the complex file data object. They also include object properties that apply to the resources in the complex file data object. The Developer tool displays overview properties for complex files in the **Overview** view.

General Properties

The following table describes the general properties that you configure for complex files:

Property	Description
Name	The name of the complex file data object.
Description	The description of the complex file data object.
Access Method	The access method for the resource. Select **Connection** to specify an HDFS connection.
Connection	The name of the HDFS connection.

Objects Properties

The following table describes the objects properties that you configure for complex files:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Description	The description of the resource.
Access Type	Indicates that you can perform read and write operations on the complex file data object. You cannot edit this property.

Compression and Decompression for Complex File Targets

You can write compressed files, specify compression formats, and decompress files. You can use compression formats such as Bzip2 and Lz4, or specify a custom compression format.

You can compress sequence files at a record level or at a block level.

For information about how Hadoop processes compressed and uncompressed files, see the Hadoop documentation.

The following table describes the compression formats:

Compression Options	Description
None	The file is not compressed.
Auto	The Data Integration Service detects the compression format of the file based on the file extension.
DEFLATE	The DEFLATE compression format that uses a combination of the LZ77 algorithm and Huffman coding.
Gzip	The GNU zip compression format that uses the DEFLATE algorithm.
Bzip2	The Bzip2 compression format that uses the Burrows–Wheeler algorithm.
Lzo	The Lzo compression format that uses the Lempel-Ziv-Oberhumer algorithm. In a streaming mapping, the compression format is LZ4. The LZ4 compression format uses the LZ77 algorithm.
Snappy	The LZ77-type compression format with a fixed, byte-oriented encoding.
Custom	Custom compression format. If you select this option, you must specify the fully qualified class name implementing the `CompressionCodec` interface in the **Custom Compression Codec** field.

Complex File Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a complex file. Select the Input transformation to edit the general, ports, targets, and run-time properties.

General Properties

The Developer tool displays general properties for complex file targets in the **Write** view.

The following table describes the general properties that you configure for complex file targets:

Property	Description
Name	The name of the complex file. This property is read-only. You can edit the name in the **Overview** view. When you use the complex file as a target in a mapping, you can edit the name in the mapping.
Description	The description of the complex file.

Ports Properties

Port properties for a physical data object include port names and port attributes such as data type and precision.

The following table describes the ports properties that you configure for complex file targets:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Description	The description of the resource.

Target Properties

The targets properties list the targets of the complex file data object.

The following table describes the target properties that you configure for complex file targets in a streaming mapping:

Property	Description
Target	The target which the complex data object writes to. You can add or remove targets.

Advanced Properties

The Developer tool displays the advanced properties for complex file targets in the Input transformation in the **Write** view.

The following table describes the advanced properties that you configure for complex file targets in a streaming mapping:

Property	**Description**
File Directory	The directory location of the complex file target. If the directory is in HDFS, enter the path without the node URI. For example, `/user/lib/testdir` specifies the location of a directory in HDFS. The path must not contain more than 512 characters. If the directory is in the local system, enter the fully qualified path. For example, `/user/testdir` specifies the location of a directory in the local system. **Note:** The Data Integration Service ignores any subdirectories and their contents.
File Name	The name of the output file. Spark appends the file name with a unique identifier before it writes the file to HDFS.
File Format	The file format. Select **Sequence** file format for target files.
Output Key Class	The class name for the output key. By default, the output key class is NullWritable.
Output Value Class	The class name for the output value. By default, the output value class is Text.
Compression Format	Optional. The compression format for binary files. Select one of the following options: - None - Auto - DEFLATE - gzip - bzip2 - LZO - Snappy - Custom
Custom Compression Codec	Required for custom compression. Specify the fully qualified class name implementing the `CompressionCodec` interface.
Sequence File Compression Type	Optional. The compression format for sequence files. Select one of the following options: - None - Record - Block

Column Projection Properties

The following table describes the columns projection properties that you configure for complex file targets:

Property	Description
Column Name	The name of the column in the source table that contains data. This property is read-only.
Type	The native data type of the resource. This property is read-only.
Enable Column Projection	Indicates that you use a schema to publish the data to the target.
Schema Format	The format in which you stream data to the target. Select one of the following formats: - XML - JSON - Avro
Schema	Specify the XSD schema for the XML format, the sample JSON for the JSON format, or sample Avro file for the Avro format.
Column Mapping	Click **View** to see the mapping of the data object to target mapping.

Complex File Execution Parameters

When you write to an HDFS complex file, you can configure how the complex file data object writes to the file. Specify these properties in the execution parameters property of the streaming mapping.

Use execution parameters to configure the following properties:

Rollover properties

When you write to an HDFS complex file, the file rollover process closes the current file that is being written to and creates a new file on the basis of file size or time. When you write to the, you can configure a time-based rollover or size-based rollover. You can use the following optional execution parameters to configure rollover:

- `rolloverTime`. You can configure a rollover of the HDFS file when a certain period of time has elapsed. Specify rollover time in hours. For example, you can specify a value of 1.
- `rolloverSize`. You can configure a rollover of the HDFS target file when the target file reaches a certain size. Specify the size in GB. For example, you can specify a value of 1.

The default is size-based rollover. You can implement both rollover schemes for a target file, in which case, the event that occurs first triggers a rollover. For example, if you set rollover time to 1 hour and rollover size to 1 GB, the target service rolls the file over when the file reaches a size of 1 GB even if the 1-hour period has not elapsed.

Pool properties

You can configure the maximum pool size that one Spark executor can have to write to a file. Use the `pool.maxTotal` execution parameter to specify the pool size. Default pool size is 8.

Retry Interval

You can specify the time interval for which Spark tries to create the target file or write to it if it fails to do so the first time. Spark tries a maximum of three times during the time interval that you specify. Use the `retryTimeout` execution parameter to specify the timeout in milliseconds. Default is 30,000 milliseconds.

HBase Data Objects

An HBase data object is a physical data object that represents data in an HBase resource. After you create an HBase connection, create an HBase data object with a write data operation to write data to an HBase table.

When you create an HBase data object, you can select an HBase table and view all the column families in the table. You can specify the column names in the column family if you know the column name and data type, or you can search the rows in the HBase table and specify the columns.

You can read write to a column family or to a single binary column. When you create the data object, specify the column families to which you can write or choose to write all the data as a single stream of binary data.

Data Object Column Configuration

When you want to write data to columns in a column family, you can specify the columns when you create the HBase data object.

You can write data to columsn in one f the following ways:

- Add the columns in the column families.
- Search for the columns names in the column family and add the columns.
- Get all the columns in a column family as a single stream of binary data.

Add Columns

When you create a data object, you can specify the columns in one or more column families in an HBase table.

When you add an HBase table as the resource for an HBase data object, all the column families in the HBase table appear. If you know the details of the columns in the column families, you can select a column family and add the column details. In the **Column Families** dialog box, select the column family to which you want to add the columns. Column details include column name, data type, precision, and scale.

Although data is stored in binary format in HBase tables, you can specify the associated data type of the column to transform the data. To avoid data errors or incorrect data, verify that you specify the correct data type for the columns.

Verify that you specify valid column details when you add columns to avoid unexpected run-time behaviors. If you add a column that does not exist in the column family and create a data object read operation, the Data Integration Service returns a null value for the column at run time. If you do not specify a value for a column when you write data to an HBase table, the Data Integration Service specifies a null value for the column at run time.

If the HBase table has more that one column family, you can add column details for multiple column families when you create the data object. Select one column family at a time and add the columns details. The column family name is the prefix for all the columns in the column family for unique identification.

Search and Add Columns

When you create a data object, you can search the rows in an HBase table to identify the column in the table and select the columns you want to add.

When you do not know the columns in an HBase table, you can search the rows in the table to identify all the columns and the occurrence percentage of the column. You can infer if the column name is valid based on the number of times the column occurs in the table. For example, if column name eName occurs rarely while column name empName occurs in a majority of rows, you can infer the column name as empName.

When you search and add columns, you can specify the maximum number of rows to search and the occurrence percentage value for a column. If you specify the maximum numbers of rows as 100 and the column occurrence percent as 90, all columns that appear at least 90 times in 100 rows appear in the results. You can select the columns in the results to add the columns to the data object.

Get All Columns

Binary data or data that can be converted to a byte array can be stored in an HBase column. You can read from and write to an HBase tables in bytes.

When you create a data object, you can choose to get all the columns in a column family as a single stream of binary data.

Use the HBase data object as a source to read data in all the columns in the column family as a single stream of binary data. Use the HBase data object as a target to write data in all the columns in the source data object as a single column of binary data in the target HBase table.

The Data Integration Service generates the data in the binary column based on the protobuf format. Protobuf format is an open source format to describe the data structure of binary data. The protobuf schema is described as messages.

HBase Object Overview Properties

The Data Integration Service uses overview properties when it writes data to an HBase resource.

Overview properties include general properties that apply to the HBase data object. They also include object properties that apply to the resources in the HBase data object. The Developer tool displays overview properties for HBase resources in the Overview view.

General Properties

The following table describes the general properties that you configure for the HBase data objects:

Property	Description
Name	Name of the HBase data object.
Location	The project or folder in the Model repository where you want to store the HBase data object.
Connection	Name of the HBase connection.

Add Column Properties

In the **Column Families** dialog box, select the column family to which you want to add the columns. The following table describes the column properties that you configure when you associate columns with column families:

Property	Description
Name	Name of the column in the column family.
Type	Data type of the column.

Property	Description
Precision	Precision of the data.
Scale	Scale of the data.

Search and Add Column Properties

The following table describes the column properties that you configure when you search for columns in column families and add the required columns:

Property	Description
Maximum rows to sample	Maximum number of rows in the HBase table you want to include while searching for columns. Default is 100.
Column occurrence percent	The threshold occurrence percentage of the column. A column appears in the results when the occurrence percentage value of the column meets or exceeds the threshold value. Default is 90.

HBase Data Object Write Properties

The Data Integration Service uses write properties when it writes data to an HBase resource.

HBase data object write operation properties include run-time properties that apply to the HBase data object.

Advanced Properties

The Developer tool displays the advanced properties for HBase targets in the **Advanced** view.

The following table describes the advanced properties that you can configure for an HBase data object in a streaming mapping:

Property	Description
Date Time Format	Format of the columns of the date data type. Specify the date and time formats by using any of the Java date and time pattern strings.
Auto Flush	Optional. Indicates whether you want to enable Auto Flush to run each Put operation immediately. You can set auto flush to the following values: - Enable **Auto Flush** to set the value to true. The Data Integration Service runs each Put operation immediately as it receives them. The service does not buffer or delay the Put operations. Operations are not retried on failure. When you enable auto flush, the operations are slow as you cannot run operations in bulk. However, you do not lose data as the Data Integration Service writes the data immediately. - Disable **Auto Flush** to set the auto flush value to false. When you disable auto flush, the Data Integration Service accepts multiple Put operations before making a remote procedure call to perform the write operations. If the Data integration Service stops working before it flushes any pending data writes to HBase, that data is lost. Disable auto flush if you need to optimize performance. Default is disabled.

JMS Data Objects

After you configure a Messaging connection, create a JMS data object to read from or write to JMS sources. You can read or write data in JSON, XML, or Avro format.

JMS providers are message-oriented middleware systems that send JMS messages. The JMS data object connects to a JMS provider to read or write data.

The JMS data object can read JMS messages from a JMS provider message queue or write JMS messages to a JMS provider. When you configure a JMS data object, configure properties to reflect the message structure of the JMS messages. The input ports and output ports are JMS message headers.

When you configure the read data operation properties, specify the format in which the JMS data object reads data. Similarly, when you configure the write data operation properties, specify the format in which the JMS data object writes data. You can specify XML, JSON, or Avro as format. When you specify XML format, you must provide an XSD file. When you specify JSON or Avro format, you must provide a sample file.

Integration with JMS

You manually create JMS source and target data objects to reflect the message structure of JMS messages.

The JMS data object can read messages of type TextMessage. This type of message contains a string object. TextMessages can contain XML, JSON, or Avro message data.

JMS Message Structure

JMS messages contain the following components:

- Header
- Properties
- Body

Header Fields

JMS messages contain a fixed number of header fields. Each JMS message uses these fields regardless of message type. Every JMS source and target definition includes a pre-defined set of header fields.

The following table describes the JMS message header fields:

Header Field	Description
JMSDestination	Destination to which the message is sent. JMS destinations can be a message queue or a recipient who listens for messages based on the message topic.
JMSDeliveryMode	Delivery mode of the message. The delivery mode can be persistent or non-persistent.
JMSMessageID	Unique identification value for the message.
JMSTimestamp	Time at which the message was handed off to the provider to be sent to the destination.
JMSCorrelationID	Links one message with another. For example, JMSCorrelationID can link a response message with the corresponding request message.
JMSReplyTo	Destination to which a reply message can be sent.

Header Field	Description
JMSRedelivered	Indicates that a message might have been delivered previously, but not acknowledged.
JMSType	Type of message based on a description of the message. For example, if a message contains a stock trade, the message type might be stock trade.
JMSExpiration	Amount of time in milliseconds the message remains valid. The messages remain in memory during this period.
JMSPriority	Priority of the message from 0-9. 0 is the lowest priority. 9 is the highest.

Property Fields

JMS source and target definitions can optionally include message property fields. Property fields contain additional message header information. JMS providers use properties in a JMS message to give provider-specific information. Applications that use a JMS provider can add property fields with application-specific information to a message.

Body Fields

JMS source and target definitions can optionally include a message body. The body contains one or more fields. Only certain types of JMS messages contain a body.

JMS Data Object Properties

The Data Integration Service uses overview properties when it reads data from a JMS source.

Overview properties include general properties that apply to the JMS data object. They also include object properties that apply to the resources in the JMS data object. The Developer tool displays overview properties for JMS messages in the Overview view.

General Properties

The following table describes the general properties that you configure for JMS data objects:

Property	Description
Name	The name of the JMS data object.
Description	The description of the JMS data object.
Connection	The name of the JMS connection.

Objects Properties

The following table describes the objects properties that you configure for JMS data objects:

Property	Description
Name	The name of the topic or topic pattern of the JMS source.
Description	The description of the JMS source.

Property	Description
Native Name	The native name of JMS source.
Path Information	The type and name of the topic or topic pattern of the JMS source.

JMS Data Object Read Properties

The Data Integration Service uses read properties when it reads data from a JMS source. You can edit the format, run-time, and advanced properties.

General Properties

The Developer tool displays general properties for JMS sources in the **Read** view.

The following table describes the general properties that you view for JMS sources:

Property	Description
Name	The name of the JMS source. This property is read-only. You can edit the name in the **Overview** view. When you use the JMS source as a source in a mapping, you can edit the name in the mapping.
Description	The description of the JMS source.

Ports Properties

Ports properties for a physical data object include port names and port attributes such as data type and precision.

The following table describes the ports properties that you configure for JMS sources:

Property	Description
Name	The name of the JMS source.
Type	The native data type of the source.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Scale	The scale of the data type.
Description	The description of the resource.

Sources Properties

The sources properties list the resources of the JMS data object. You can add or remove resources in the data object.

Run-time Properties

The run-time properties displays the name of the connection.

The following table describes the run-time property that you configure for JMS sources:

Property	Description
Connection	Name of the JMS connection.

Advanced Properties

The Developer tool displays the advanced properties for JMS sources in the Output transformation in the **Read** view.

You can configure the following advanced properties for JMS sources:

Property	Description
JMS Destination	Name of the queue or topic to which the JMS provider publishes messages. The data object subscribes to JMS messages from this queue or topic.
Client ID	Client identifier to identify the connection and set up a durable connections.
Durable Subscription Name	Name of the durable subscription that can receive messages sent while the subscribers are not active. Durable subscriptions provide the flexibility and reliability of queues, but still allow clients to send messages to many recipients.
JMS Message Selector	Criteria for filtering message header or message properties, to limit which JMS messages the data object receives.
Guaranteed Processing	Guaranteed processing ensures that the mapping processes messages published by the sources and delivers them to the targets at least once. In the event of a failure, there could be potential duplicates but the messages are processed successfully. If the external source or the target is not available, the mapping execution stops to avoid any data loss. Select this option for guaranteed delivery of data streamed from the JMS sources.

Column Projections Properties

The following table describes the columns projection properties that you configure for JMS sources:

Property	Description
Column Name	The name field that contains data. This property is read-only.
Type	The native data type of the resource. This property is read-only.
Enable Column Projection	Indicates that you use a schema to read the data that the source streams.
Schema Format	The format in which the source streams data. You can select one of the following formats: - XML - JSON - Avro
Schema	Specify the XSD schema for the XML format, the sample JSON for the JSON format, or the sample Avro file for the Avro format.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.

JMS Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a JMS source. You can edit the format, run-time, and advanced properties.

General Properties

The Developer tool displays general properties for JMS targets in the **Write** view.

The following table describes the general properties that you view for JMS targets:

Property	Description
Name	The name of the JMS target. This property is read-only. You can edit the name in the **Overview** view. When you use the JMS target as a target in a mapping, you can edit the name in the mapping.
Description	The description of the JMS target.

Ports Properties

Ports properties for a physical data object include port names and port attributes such as data type and precision.

The following table describes the ports properties that you configure for JMS targets:

Property	Description
Name	The name of the JMS target.
Type	The native data type of the target.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Scale	The scale of the data type.
Description	The description of the resource.

Target Properties

The target properties list the resources of the JMS data object. You can add or remove resources in the data object.

Run-time Properties

The run-time properties displays the name of the connection.

The following table describes the run-time property that you configure for JMS targets:

Property	Description
Connection	Name of the JMS connection.

Advanced Properties

The Developer tool displays the advanced properties for JMS targets in the Input transformation in the **Write** view.

You can configure the following advanced properties for JMS targets:

Property	Description
Destination	Name of the queue or topic to which the JMS provider publishes messages. The data object subscribes to JMS messages from this queue or topic.
Destination Type	The type of destination to which the JMS provider publishes messages. Select one of the following destination types: - Topic - Queue

Property	Description
Payload Type	The format in which the JMS provider writes messages. Specify one of the following format of the data can be one of the following types: - JSON - XML - Avro
Delivery Mode	The delivery mode of a message. Specify one of the following modes: - 1. Indicates that the delivery mode is NON-PERSISTENT. A delivery mode of 1 indicates that the data is not persisted in the broker. Messages sent in a NON-PERSISTENT mode are lost if the broker goes down while the messages are being sent. - 2. Indicates that the delivery mode is PERSISTENT MODE. To persist messages in the broker set the message delivery mode to 2. Messages that are persisted in the broker are available when the broker goes down.

Kafka Data Objects

After you configure a Messaging connection, create a Kafka data object to read from or write to Apache Kafka brokers. You can read or write data in JSON, XML, or Avro format.

Kafka runs as a cluster comprised of one or more servers each of which is called a broker. Kafka brokers stream data in the form of messages. These messages are published to a topic. When you configure the Kafka data object, specify the name of the topic name that you read from. Similarly, when you write data to a Kafka messaging stream, specify the name of the topic name that you publish to.

Kafka topics are divided into partitions. Spark Streaming can read the partitions of the topics in parallel. This gives better throughput and could be used to scale the number of messages processed. Message ordering is guaranteed only within partitions. For optimal performance you should have multiple partitions. When you write to Kafka brokers, you can use the `partionId`, `Key`, and `TopicName` output ports. You can override these ports when you create the mapping.

You can create or import a Kafka data object. When you create a Kafka data object, a read and write operation is created. You can use the Kafka data object read operation as a source and the Kafka data object write operation as a target in streaming mappings. If you want to configure high availability for the mapping, ensure that the Kafka cluster is highly available.

When you configure the data operation properties, specify the format in which the Kafka data object reads or writes data. You can specify XML, JSON, or Avro as format. When you specify XML format, you must provide an XSD file. When you specify JSON or Avro format, you must provide a sample file.

For more information about Kafka clusters, Kafka brokers, and partitions see http://kafka.apache.org/082/documentation.html.

Kafka Data Object Overview Properties

The Data Integration Service uses overview properties when it reads data from or writes data to a Kafka broker.

Overview properties include general properties that apply to the Kafka data object. They also include object properties that apply to the resources in the Kafka data object. The Developer tool displays overview properties for Kafka messages in the Overview view.

General Properties

The following table describes the general properties that you configure for Kafka data objects:

Property	Description
Name	The name of the Kafka data object.
Description	The description of the Kafka data object.
Connection	The name of the Kafka connection.

Objects Properties

The following table describes the objects properties that you configure for Kafka data objects:

Property	Description
Name	The name of the topic or topic pattern of the Kafka broker.
Description	The description of the Kafka broker.
Native Name	The native name of Kafka broker.
Path Information	The type and name of the topic or topic pattern of the Kafka broker.

Column Properties

The following table describes the column properties that you configure for Kafka data objects:

Property	Description
Name	The name of the target.
Native Name	The native name of the target.
Type	The native data type of the target.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Scale	The scale of the data type.
Description	The description of the target.

Kafka Data Object Read Properties

The Data Integration Service uses read properties when it reads data from a Kafka broker.

General Properties

The Developer tool displays general properties for Kafka sources in the **Read** view.

The following table describes the general properties that you view for Kafka sources:

Property	Description
Name	The name of the Kafka broker. This property is read-only. You can edit the name in the **Overview** view. When you use the Kafka broker as a source in a mapping, you can edit the name in the mapping.
Description	The description of the Kafka broker.

Ports Properties

Ports properties for a physical data object include port names and port attributes such as data type and precision.

The following table describes the ports properties that you configure for Kafka broker sources:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Scale	The scale of the data type.
Description	The description of the resource.

Run-time Properties

The run-time properties displays the name of the connection.

The following table describes the run-time property that you configure for Kafka sources:

Property	Description
Connection	Name of the Kafka connection.

Advanced Properties

The Developer tool displays the advanced properties for Kafka sources in the Output transformation in the Read view.

The following table describes the advanced property that you can configure for Kafka sources:

Property	Description
Guaranteed Processing	Guaranteed processing ensures that the mapping processes messages published by the sources and delivers them to the targets at least once. In the event of a failure, there could be potential duplicates but the messages are processed successfully. If the external source or the target is not available, the mapping execution stops to avoid any data loss. Select this option to avoid data loss in the event of failure of Kafka brokers.

Sources Properties

The sources properties list the resources of the Kafka data object.

The following table describes the sources property that you can configure for Kafka sources:

Property	Description
Sources	The sources which the Kafka data object reads from. You can add or remove sources.

Column Projection Properties

The Developer tool displays the column projection properties in the Properties view of the Read operation.

To specify column projection properties, double click on the read operation and select the data object. The following table describes the columns projection properties that you configure for Kafka sources:

Property	Description
Column Name	The name field that contains data. This property is read-only.
Type	The native data type of the source. This property is read-only.
Enable Column Projection	Indicates that you use a schema to read the data that the source streams.
Schema Format	The format in which the source streams data. Select one of the following formats: - XML - JSON - Avro
Schema	Specify the XSD schema for the XML format or the sample file for JSON or Avro format.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.

Kafka Data Object Write Properties

The Data Integration Service uses write properties when it writes data to a Kafka broker.

General Properties

The Developer tool displays general properties for Kafka targets in the **Write** view.

The following table describes the general properties that you view for Kafka targets:

Property	Description
Name	The name of the Kafka broker. This property is read-only.
Description	The description of the Kafka broker.

Ports Properties

Ports properties for a physical data object include port names and port attributes such as data type and precision.

The following table describes the ports properties that you configure for Kafka broker sources:

Property	Description
Name	The name of the resource.
Type	The native data type of the resource.
Precision	The maximum number of significant digits for numeric data types, or the maximum number of characters for string data types.
Scale	The scale of the data type.
Description	The description of the resource.

Run-time Properties

The run-time properties displays the name of the connection.

The following table describes the run-time property that you configure for Kafka targets:

Property	Description
Connection	Name of the Kafka connection.

Target Properties

The targets properties list the targets of the Kafka data object.

The following table describes the sources property that you can configure for Kafka targets:

Property	Description
Target	The target which the Kafka data object writes to. You can add or remove targets.

Advanced Properties

The Developer tool displays the advanced properties for Kafka targets in the Input transformation in the **Write** view.

The following table describes the advanced properties that you configure for Kafka targets:

Property	Description
Metadata Fetch Timeout in milliseconds	The time after which the metadata is not fetched.
Batch Flush Time in milliseconds	The interval after which the data is published to the target.
Batch Flush Size in bytes	The batch size of the events after which the data is written to the target.

For more information about Kafka broker properties, see http://kafka.apache.org/082/documentation.html.

Column Projections Properties

The Developer tool displays the column projection properties in the **Properties** view of the Write operation.

To specify column projection properties, double click on the write operation and select the data object. The following table describes the columns projection properties that you configure for Kafka targets:

Property	Description
Column Name	The field in the target that the data object writes to. This property is read-only.
Type	The native data type of the target. This property is read-only.
Enable Column Projection	Indicates that you use a schema to publish the data to the target.
Schema Format	The format in which you stream data to the target. You can select one of the following formats: - XML - JSON - Avro

Property	Description
Schema	Specify the XSD schema for the XML format or the sample file for JSON or Avro format.
Column Mapping	The mapping of data object to the target. Click **View** to see the mapping.

Relational Data Objects

Create a relational data object with a Hive connection to write to Hive tables.

When you create a relational data object, you can add read and write operations. To use the relational data object as a target in streaming mappings, configure the relational data object write operation properties. You can select the mapping environment and run the mappings on the Spark engine of the Hadoop environment.
Note: If you enable the **Truncate target table** property in the Advanced properties while writing to the Hive table, the data in the table is overwritten. If you do not select this property, data is appended.

For more information about relational data objects and Hive connections, see the *Informatica PowerExchange for Hive User Guide*.

Relational Data Object Overview Properties

The Data Integration Service uses overview properties when it writes data to a relational data object.

The Overview properties include general properties that apply to the relational data object. They also include column properties that apply to the resources in the relational data object.

General Properties

The following table describes the general properties that you configure for relational data objects:

Property	Description
Name	Name of the relational data object.
Description	Description of the relational data object.
Connection	Name of the relational connection.

Column Properties

The following table describes the column properties that you can view for relational data objects:

Property	Description
Name	Name of the column.
Native Type	Native data type of the column.
Precision	Maximum number of significant digits for numeric data types, or maximum number of characters for string data types. For numeric data types, precision includes scale.

Property	Description
Scale	Maximum number of digits after the decimal point for numeric values.
Description	Description of the column.

Advanced Properties

Advanced properties include run-time and other properties that apply to the relational data object. The Developer tool displays advanced properties for relational data object in the **Advanced** view.

Property	Description
Connection	Name of the Hive connection.
Owner	Name of the resource owner. This property is not applicable for Hive sources and targets.
Resource	Name of the resource.
Database Type	Type of the source. This property is read-only.
Resource Type	Type of the resource. This property is read-only.

Relational Data Object Write Properties

The Data Integration Service uses write properties when it to write data to Hive..

The data object operation properties include general, ports, run-time, target, and advanced properties.

General Properties

The general properties for the write transformation include the properties for name, description, and metadata synchronization.

The following table describes the general properties that you configure for the relational data object:

Property	Description
Name	Name of the relational data object. This property is read-only. You can edit the name in the **Overview** view. When you use the relational file as a source in a mapping, you can edit the name within the mapping.
Description	Description of the relational data object.
When column metadata changes	Indicates whether object metadata is synchronized with the source. Select one of the following options: - Synchronize output ports. The Developer tool reimports the object metadata from the source. - Do not synchronize. Object metadata may vary from the source.

Ports Properties

Ports properties include column names and column attributes such as data type and precision.

The following table describes the ports properties that you configure for relational targets:

Property	Description
Name	Name of the column.
Type	Native data type of the column.
Precision	Maximum number of significant digits for numeric data types, or maximum number of characters for string data types. For numeric data types, precision includes scale.
Scale	Maximum number of digits after the decimal point for numeric values.
Description	Description of the column.
Column	Name of the column in the resource.
Resource	Name of the resource.

Run-time Properties

The run-time properties displays the connection name and reject file and directory.

The following table describes the run-time properties that you configure for Hive targets:

Property	Description
Connection	Name of the Hive connection.
Owner	Name of the Hive database.
Resource	Name of the resource.

Target Properties

The target properties lists the resource that is used in the relational data object and the target details for the resource.

The following table describes the target properties that you configure for Hive targets in a streaming mapping:

Property	Description
Target	The target which the relational data object writes to. You can add or remove targets.

Advanced Properties

The advanced properties includes the write properties used to write data to the target.

The following table describes the advanced properties that you configure for Hive targets:

Property	Description
Tracing level	Controls the amount of detail in the mapping log file.
Truncate target table	Truncates the target before loading data. **Note:** If the mapping target is a Hive partition table, you can choose to truncate the target table only with Hive version 0.11.

Sample Files

The data objects in a streaming mapping read and write data in XML, JSON, and Avro format. The sample flat files contain sample XML schema definition, sample JSON file, and sample Avro files.

Sample XSD File

When you configure the data operation properties, specify the format in which the data object reads or writes data. When you specify XML format, provide an XSD.

The following sample XSD describes the elements in an XML file:

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Tweet">
    <xs:complexType>
      <xs:sequence>
        <xs:element type="xs:string" name="customer_id"/>
        <xs:element type="xs:string" name="customer_name"/>
       </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Sample JSON File

When you configure the data operation properties, specify the format in which the data object reads or writes data. When you specify JSON format, provide a sample JSON file.

The following file is a sample JSON file:

```
{"Country":"US","Language":"EN","TweetDataVolume":"High"}
```

Sample Avro File

When you configure the data operation properties, specify the format in which the data object reads or writes data. When you specify Avro format, provide a sample Avro file.

The following file is a sample Avro file:

```
{
  "type" : "record",
  "name" : "twitter_schema",
  "namespace" : "com.miguno.avro",
  "fields" : [ {
    "name" : "username",
    "type" : "string",
    "doc"  : "Name of the user account on Twitter.com"
  }, {
    "name" : "tweet",
    "type" : "string",
    "doc"  : "The content of the user's Twitter message"
  }, {
    "name" : "timestamp",
    "type" : "long",
    "doc"  : "Unix epoch time in seconds"
  } ],
  "doc:" : "A basic schema for storing Twitter messages"
}
```

CHAPTER 5

Intelligent Streaming Mappings

This chapter includes the following topics:

Intelligent Streaming Mappings Overview

Use the Developer tool to create and run Intelligent Streaming mappings in the Hadoop run-time environment and process data that is in JSON, XML, or Avro format.

When you create a streaming mapping, select the Hadoop environment and the Spark engine. When you run a streaming mapping, the Data Integration Service pushes the processing to nodes on a Spark engine in the Hadoop cluster.

When you configure the mapping, you can specify the following configurations:

- Mapping configuration. The Developer tool uses this configuration when you run mappings through the Run dialog box or from the command line. You must configure a Messaging connection for the mapping and the run-time properties for the Hadoop environment. You can configure the Spark engine properties in the Hadoop connection. You can also use parameters to represent properties in the Hadoop environment if you need to use constant values between mapping runs.
- Run configuration. These configurations apply to mappings that you run through the Developer tool. You can configure the source properties for streaming mappings.

When you run the mapping, the Data Integration Service converts the mapping to a Scala program and package it in a JAR file and sends it to the Hadoop cluster. You can view the details in the Spark execution plan in the Developer tool or Administrator tool.

Mapping Configurations

To configure a mapping, configure the connection and run-time properties for the mapping.

When you configure the mapping, configure the following run-time properties:

Validation Environment

The environment in which the validations are done. Select Hadoop in the validation environment and select the Spark engine. The Data Integration Service pushes the mapping logic to the Spark engine..

Execution Environment

The environment in which the mappings are executed. Select Hadoop as the execution environment.

Hadoop

Specify the following properties for the Spark engine:

- Connection. Select the connection to the Spark engine used for pushdown of processing. Select **Connection** and browse for a connection or select a connection parameter.
- Runtime Parameters. An optional list of configuration parameters to apply to the Spark engine. You can change the default Spark configuration properties values, such as `spark.executor.memory` or `spark.driver.cores`.
  Use the following format:

  ```
  <property1>=<value>
  ```

  •<property1> is a Spark configuration property.

  •<value> is the value of the property.

  To enter multiple properties, separate each name-value pair with the following text: `&:`.

  If you use a JMS source in the mapping, configure two or more executors for the mapping. For example, use the following configuration:

  ```
  spark.executor.instances=2 &: spark.executor.cores=2 &: spark.driver.cores=1
  ```

Source Configuration

Specify the following properties to configure how the data is processed:

- Maximum Rows Read. Specify the maximum number of rows that are read before the mapping stops running. Default is `Read All Rows`.
- Maximum Runtime Interval. Specify the maximum time to run the mapping before it stops. If you set values for this property and the Maximum Rows Read property, the mapping stops running after one of the criteria is met. Default is `Run Indefinitely`. A value of `Run Indefinitely` enables the mapping to run without stopping.
- State Store. Specify the HDFS location on the cluster to store information about the state of the Spark Job. Default is `<Home Directory>/stateStore`
  You can configure the state store as part of the configuration of execution options for the Data Integration Service.

You can use these properties to test the mapping.

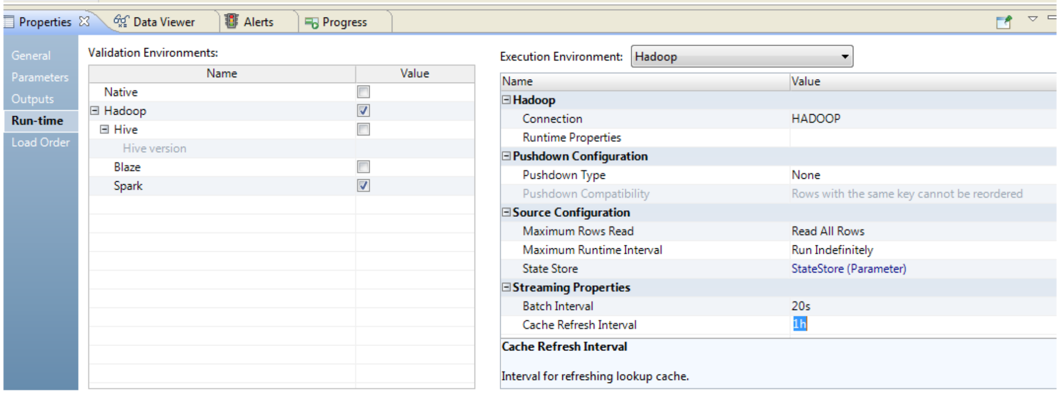
Streaming Properties

Specify the following streaming properties:

- Batch interval. The Spark engine processes the streaming data from sources and publishes the data in batches. The batch interval is number of seconds after which a batch is submitted for processing.

- Cache refresh interval. You can cache a large lookup source or small lookup tables. When you cache the lookup source, the Data Integration Service queries the lookup cache instead of querying the lookup source for each input row. You can configure the interval for refreshing the cache used in a relational Lookup transformation.

The following image shows the connection and run-time properties:



Validation Environments:

Name	Value
Native	□
⊟ Hadoop	☑
⊟ Hive	□
Hive version	
Blaze	□
Spark	☑

Execution Environment: Hadoop

Name	Value
⊟ **Hadoop**	
Connection	HADOOP
Runtime Properties	
⊟ **Pushdown Configuration**	
Pushdown Type	None
Pushdown Compatibility	Rows with the same key cannot be reordered
⊟ **Source Configuration**	
Maximum Rows Read	Read All Rows
Maximum Runtime Interval	Run Indefinitely
State Store	StateStore (Parameter)
⊟ **Streaming Properties**	
Batch Interval	20s
Cache Refresh Interval	1h

Cache Refresh Interval

Interval for refreshing lookup cache.

Run Configurations

The Developer tool applies configuration properties when you run streaming mappings. Set configuration properties for streaming mappings in the **Run** dialog box.

Configure the following source properties:

- Read all rows. Reads all rows from the source.
- Read up to how many rows. The maximum number of rows to read from the source if you do not read all rows.
- Maximum runtime interval. The maximum time to run the mapping before it stops. If you set values for this property and the **Maximum Rows Read** property, the mapping stops running after one of the criteria is met.

Transformations in a Streaming Mapping

Informatica Developer provides a set of transformations that perform specific functions. Some restrictions and guidelines apply to processing transformations in a Streaming mapping.

The following table describes rules and guidelines for the transformations that are supported in a Streaming mapping:

Transformation	Rules and Guidelines
Aggregator	Mapping validation fails in the following situations: - The transformation contains stateful variable ports. - The transformation contains unsupported functions in an expression.
Expression	Mapping validation fails in the following situations: - The transformation contains stateful variable ports. - The transformation contains unsupported functions in an expression. If an expression results in numerical errors, such as division by zero or SQRT of a negative number, it returns an infinite or an NaN value. In the native environment, the expression returns null values and the rows do not appear in the output.
Filter	Supported without restrictions.
Java	The following restrictions apply to the Java transformation: - The value Transaction for transformation scope is not valid. - The transformation is always Stateless - The Partitionable field is ignored.
Joiner	Mapping validation fails in the following situations: - Case sensitivity is disabled.
Lookup	Use a Lookup transformation to look up data in a flat file, HDFS, Hive, or Sqoop. Mapping validation fails in the following situations: - Case sensitivity is disabled. - The transformation is not configured to return all rows that match the condition. - The lookup is a data object. - The cache is configured to be shared, named, persistent, dynamic, or uncached. The cache must be a static cache. The mapping fails in the following situations: - The transformation is unconnected. You cannot use a float data type to look up data in a Hive table as comparing equality of floating point numbers is unsafe. To use a Lookup transformation on Sqoop in a Cloudera distribution, perform the following configuration: 1. In the Yarn configuration, locate the property `NodeManager Advanced Configuration Snippet (Safety Valve) for mapred-site.xml` 2. Add the following xml snippet: `<property> <name>mapreduce.application.classpath</name> <value> $HADOOP_MAPRED_HOME/,$HADOOP_MAPRED_HOME/lib/, $MR2_CLASSPATH</value> </property>` **Note:** Informatica recommends that you select the **Ignore null values that match** property in Lookup transformation advanced properties to avoid cross join of DataFrames.
Router	Supported without restrictions.

Transformation	Rules and Guidelines
Sorter	Mapping validation fails in the following situations: - Case sensitivity is disabled. The Data Integration Service logs a warning and ignores the Sorter transformation in the following situations: - There is a type mismatch in between the target and the Sorter transformation sort keys. - The transformation contains sort keys that are not connected to the target. - The Write transformation is not configured to maintain row order. - The transformation is not directly upstream from the Write transformation. The Data Integration Service treats null values as high even if you configure the transformation to treat null values as low.
Union	Supported without restrictions.
Window	Supported without restrictions. See the Window Transformation chapter in this guide for more information.
Transformations not listed in this table are not supported.	

For more information about the transformations, see the *Informatica Developer Transformation Guide*.

For more information about restrictions on the Spark engine, see the *Informatica Big Data Management User Guide*.

Rules in a Streaming Mapping

A rule expresses the business logic that defines conditions applied to source data. You can add expression rules to streaming mapping to cleanse, change, or validate data. Create expression rules in the Analyst tool.

You might want to use a rule in different circumstances. You can add a rule to cleanse one or more data columns.

Rules that you create in the Analyst tool appear as mapplets in the Developer tool. You can use a mapplet in a mapping or validate the mapplet as a rule.

In a streaming mapping, you can use the following functions in expression rules:

- LastDay
- Add to Date
- Concat
- Date Diff
- Date Time
- Greatest
- Choose
- Least
- Length
- Null

- Reverse
- Truncate
- Convert to Data

For more information about rules, see the *Informatica Profile Guide*.

For more information about mapplets, see the *Informatica Mapping Guide*.

Mapping Validation

When you develop a streaming mapping, you must configure it so that the Data Integration Service can read and process the entire mapping. The Developer tool marks a mapping as not valid when it detects errors that will prevent the Data Integration Service from running the mapping.

The Developer tool considers the following types of validation:

- Environment
- Data object
- Transformation
- Run-time

Environment Validation

The Developer tool performs environment validation each time you validate a streaming mapping.

The Developer tool generates an error in the following scenarios:

- The mapping has Native environment.
- The mapping does not have a Spark validation environment and Hadoop execution environment.
- The mapping has a Hadoop execution environment and has Native, Blaze, Hive on MapReduce, and Spark validation environment.

Data Object Validation

When you validate a mapping, the Developer tool verifies the source and target data objects that are part of the streaming mapping.

The Developer tool generates an error in the following scenarios:

- The mapping contains a source data object other than Kafka or JMS, and target data object other than Kafka, complex file, or relational data object.
- The read and write properties of the Kafka, JMS, and complex file data objects are not specified correctly.
- If you add a Kafka or a JMS data object to an LDO mapping, REST mapping, or a mapplet.

Transformation Validation

When you validate a mapping, the Developer tool performs validation on the transformations that are part of the streaming mapping.

The Developer tool performs the following validations:

- A mapping cannot contain a transformation other than the Aggregator , Expression, Filter , Joiner, Lookup, Router, Union, and Window transformations.
- A Window transformation is added between a streaming source and a Sorter, Aggregator, or Joiner transformation.
- A Window transformation has at least one upstream streaming source.
- All Window transformations have a slide interval that is a multiple of the mapping batch interval.
- A Window transformation that is downstream from another Window transformation must have a slide interval that is a multiple of the slide interval of the upstream Window transformation.
- The slide interval of a sliding Window transformation must be less than window size.
- The format of the parameter of the window size must have the TimeDuration parameter type.
- The window size and slide interval of a Window transformation must be greater than 0.
- The downstream Window transformation in the pipelines leading to a Joiner transformation must have the same slide intervals.
- A Window transformation cannot be added to a Logical Data Object mapping, REST mapping, or a mapplet.
- If one pipeline leading to a Union transformation has a Window transformation, all streaming pipelines must have a Window transformation. All downstream Window transformations in the pipelines leading to the Union transformations must have the same slide interval.
- A Union transformation cannot be used to merge data from streaming and non-streaming pipelines.
- A Union transformation does not require a Window transformation between a streaming source and itself.

Run-time Validation

The Developer Tool performs validations each time you run a streaming mapping.

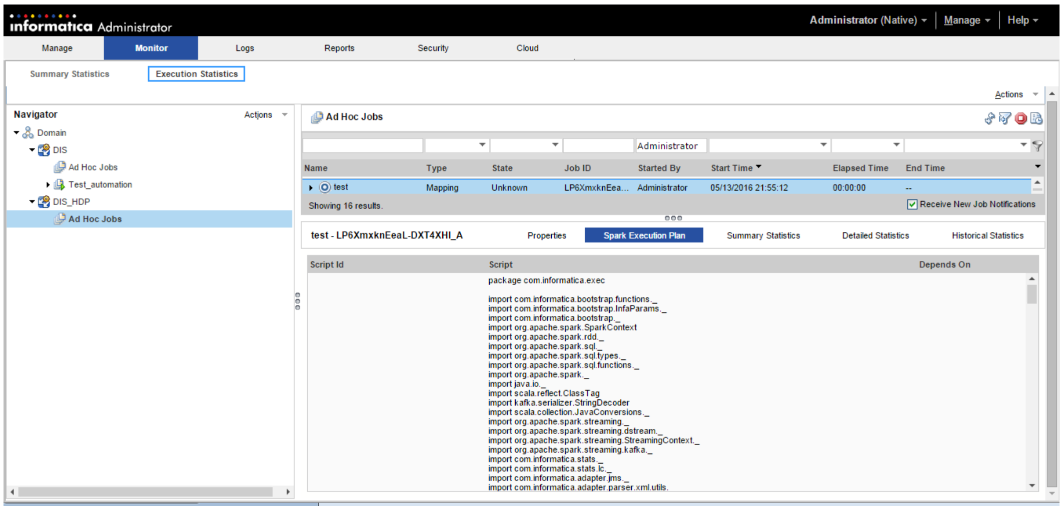
The Developer tool generates an error in the following scenarios:

- The state store is not configured when you specify the source configuration properties for the mapping.
- Either the Maximum Rows Read or the Maximum Runtime Interval property is not configured when you specify the source configuration properties for the mapping.
- The Maximum Runtime Interval does not have the correct time format.
- The Batch Interval does not have the correct time format.
- If the default value of the Batch Interval and Maximum Runtime Interval properties are not specified.

Monitor Jobs

You can monitor statistics and view log events for a streaming mapping job in the **Monitoring** tab of the Administrator tool.

The following image shows the **Monitor** tab in the Administrator tool:



Name	Type	State	Job ID	Started By	Start Time	Elapsed Time	End Time
test	Mapping	Unknown	LP6XmxknEea...	Administrator	05/13/2016 21:55:12	00:00:00	--

Showing 16 results.

☑ Receive New Job Notifications

test - LP6XmxknEeaL-DXT4XHI_A

Script Id	Script	Depends On

```
package com.informatica.exec

import com.informatica.bootstrap.functions._
import com.informatica.bootstrap.InfaParams._
import com.informatica.bootstrap._
import org.apache.spark.SparkContext
import org.apache.spark.rdd._
import org.apache.spark.sql._
import org.apache.spark.sql.types._
import org.apache.spark.sql.functions._
import org.apache.spark._
import java.io._
import scala.reflect.ClassTag
import kafka.serializer.StringDecoder
import scala.collection.JavaConversions._
import org.apache.spark.streaming._
import org.apache.spark.streaming.dstream._
import org.apache.spark.streaming.StreamingContext._
import org.apache.spark.streaming.kafka._
import com.informatica.stats._
import com.informatica.stats.lc._
import com.informatica.adapter.jms._
import com.informatica.adapter.parser.xml.utils.
```

Use the **Execution Statistics** view of the **Monitor** tab to monitor properties, run-time statistics, and run-time reports.

The **Spark Execution Plan** view appears when you run a streaming mapping with the Spark engine in the Hadoop environment and displays the execution plan for the Spark engine mapping.

View the streaming mapping statistics in the **Spark Execution Plan** view of the **Execution Statistics** view.

Note: If a failover occurs, the statistics might not be accurate.

High Availability

You can configure the Data Integration Service and the Developer tool to run streaming mappings in a highly available Hadoop cluster. You can configure high availability for the streaming mapping.

Before you configure high availability for the mapping, verify the following prerequisites:

- The Hadoop cluster must be highly available.
- The Informatica domain, the Data Integration Service, and the Model Repository Service should be highly available.
- If the mapping has Kafka sources in it, the Kafka cluster should be highly available.

For information about Informatica domain high availability, see the *Informatica Administrator Guide*.

For information about Spark high availability, see the *Informatica Big Data Management User Guide*.

High Availability Configuration

To configure high availability for the streaming mapping, configure a state store directory for the source and guaranteed processing of the messages streamed by the source. Also configure the Spark execution parameters to enable the mapping to run without failing.

To configure high availability, perform the following configurations:

State store configuration

Configure a state store directory. Spark uses the state store directory to store the checkpoint information at regular intervals during the execution of the mapping. If a failure occurs, Spark restarts processing by reading from this state store directory.

Execution parameters

To ensure that the mapping runs without failing, configure the maximum number of tries to submit the mapping to Spark for processing. Configure the `spark.yarn.maxAppAttempts` and `yarn.resourcemanager.am.max-attempts` execution parameters when you configure the mapping properties. The values that you specify for both parameters must be equal and less than the values configured on the CDH or HortonWorks configuration.

Troubleshooting Streaming Mappings

When I run a streaming mapping, the mapping fails, and I see the following errors in the application logs of the Hadoop cluster:

```
User class threw exception: org.apache.spark.SparkException: Job aborted due to stage
failure:
Task 0 in stage 1.0 failed 4 times, most recent failure: Lost task 0.3 in stage 1.0 (TID
4, localhost):
java.lang.Exception: Retry Failed: Total 3 attempts made at interval 10000ms
at
com.informatica.adapter.streaming.hdfs.common.RetryHandler.errorOccured(RetryHandler.java
:74)
at
com.informatica.adapter.streaming.hdfs.HDFSMessageSender.sendMessages(HDFSMessageSender.j
ava:55)
at com.informatica.bootstrap.InfaStreaming$$anonfun$writeToHdfsPathRealtime$1$$anonfun
$apply$5.apply(InfaStreaming.scala:144)
at com.informatica.bootstrap.InfaStreaming$$anonfun$writeToHdfsPathRealtime$1$$anonfun
$apply$5.apply(InfaStreaming.scala:132)
at org.apache.spark.rdd.RDD$$anonfun$foreachPartition$1$$anonfun$apply
$28.apply(RDD.scala:902)
at org.apache.spark.rdd.RDD$$anonfun$foreachPartition$1$$anonfun$apply
$28.apply(RDD.scala:902)
at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1916)
at org.apache.spark.SparkContext$$anonfun$runJob$5.apply(SparkContext.scala:1916)
at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:70)
at org.apache.spark.scheduler.Task.run(Task.scala:86)
at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:274)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at java.lang.Thread.run(Thread.java:745)
```

This error occurs if the HDFS NameNode is configured incorrectly.

To resolve this error, ensure that you specify the NameNode URI correctly in the HDFS connection and that the NameNode is up and running.

When I try to run streaming mappings concurrently, a few of the mappings fail and I get the following error in the Data Integration Service logs:

```
Caused by: java.lang.OutOfMemoryError: Java heap space
at java.util.Arrays.copyOf(Arrays.java:3332)
at java.lang.AbstractStringBuilder.ensureCapacityInternal(AbstractStringBuilder.java:124)
at java.lang.AbstractStringBuilder.append(AbstractStringBuilder.java:448)
at java.lang.StringBuilder.append(StringBuilder.java:136)
```

This error occurs when the Data Integration Service does not have sufficient memory to run concurrent mappings. The Data Integration Service logs are located at `<INFA_HOME>/logs/<node name>/services/DataIntegrationService/disLogs/`

To resolve this error, configure the following advanced properties of the Data Integration Service:

- Maximum Heap Size. Specify a minimum value of 2048M. Default is 640M.
- JVM command Line Options. Specify a minimum value of 1024M for the **XX:MaxMetaspaceSize** attribute. Default is 192M.

The streaming mapping execution fails with the following error in the in the application logs of the Hadoop cluster:

```
Cleaning up the staging area /tmp/hadoop-yarn/staging/cloudqa/.staging/
job_1475754687186_0406
PriviledgedActionException as:cloudqa (auth:PROXY) via yarn (auth:SIMPLE)
cause:org.apache.hadoop.security.AccessControlException:
Permission denied: user=cloudqa, access=EXECUTE, inode="/tmp/hadoop-yarn/
staging":yarn:supergroup:drwx------
at
org.apache.hadoop.hdfs.server.namenode.DefaultAuthorizationProvider.checkFsPermission(Def
aultAuthorizationProvider.java:281)
at
org.apache.hadoop.hdfs.server.namenode.DefaultAuthorizationProvider.check(DefaultAuthoriz
ationProvider.java:262)
at
org.apache.hadoop.hdfs.server.namenode.DefaultAuthorizationProvider.checkTraverse(Default
AuthorizationProvider.java:206)
at
org.apache.hadoop.hdfs.server.namenode.DefaultAuthorizationProvider.checkPermission(Defau
ltAuthorizationProvider.java:158)
at
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionCh
ecker.java:152)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkPermission(FSNamesystem.java:
6621)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkPermission(FSNamesystem.java:
6603)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkOwner(FSNamesystem.java:6522)
```

This error occurs when a YARN user, Spark engine user, or mapping impersonation user does not have sufficient permission on the `/tmp/hadoop-yarn/staging` directory. Assign required permissions and run the mapping again.

The streaming mapping execution fails with the following error in the in the application logs of the Hadoop cluster:

```
Mapping execution fails with error "Error: : Unsupported major.minor version 52.0"
```

This error occurs if there is a mismatch in the JDK version on which Cloudera processes are running and the JDK version specified in `jdk_home` directory.

To resolve this error, ensure that both the versions are set to the version supported by Informatica.

To configure the `jdk_home` directory, perform the following steps:

1. Find the `hadoopEnv.properties` in the following directory:
   `<INFA_HOME>/services/shared/hadoop/<Hadoop distribution>/infaConf`
2. Set the `jdk_home` property correctly.

Troubleshooting Streaming Mappings with Lookup Transformation

I use Sqoop as a Lookup transformation in the streaming mapping. The mapping fails, and I see the following error in the application logs of the CDH cluster:

```
Error: Could not find or load main class org.apache.hadoop.mapreduce.v2.app.MRAppMaster
```

This error occurs if the MapReduce directory is configured incorrectly.

To resolve this error, perform the following steps:

1. In the Yarn configuration, find the `NodeManager Advanced Configuration Snippet (Safety Valve) for mapred-site.xml` property.
2. Add the following xml snippet:

   ```
   <property> <name>mapreduce.application.classpath</name> <value>$HADOOP_MAPRED_HOME/,
   $HADOOP_MAPRED_HOME/lib/,
    $MR2_CLASSPATH</value> </property>
   ```
3. Restart the affected services as indicated by Cloudera Manager and run the mapping again.

I use Sqoop as a Lookup transformation in the streaming mapping. The mapping validation fails, and I see the following errors in the Developer tool:

```
Mapping1    Mapping    The transformation [output] contains a binary data type which you
cannot use in a Streaming mapping. Use a valid data type.

Mapping1 Mapping The transformation [output] contains a binary data type which you
cannot use in a Streaming mapping. Use a valid data type.
[ID:BINARY_FIELD_NOT_SUPPORTED_STREAMING]
Lookup_ORACLE_TEST_CHAR    MRS/Sqoop_test/Mapping1

ORACLE_TEST_CHAR Relational Data Object  In relational column [TEST_NUMBER] with native
datatype [decimal], the
scale [-127] is not valid. [ID:INVALID_SCALE]  TEST_NUMBER MRS/Sqoop_test/
ORACLE_TEST_CHAR
```

The errors occur if the Lookup transformation has a data type, such as binary, that Spark Streaming or Sqoop import does not support.

To resolve this error, delete the columns of the unsupported data type in the Lookup transformation and then validate the mapping.

For more information about data type support, see the *Informatica Big Data Management User Guide*.

I use Sqoop as a Lookup transformation in the streaming mapping. The mapping fails, and the following error appears in the application logs of the Hadoop cluster:

```
User class threw exception: java.util.concurrent.ExecutionException:
java.lang.IllegalArgumentException:
/opt/cloudera/parcels/CDH-5.8.0-1.cdh5.8.0.p0.42/bin/../lib/hadoop-yarn/bin/yarn:
```

```
line 318: /usr/java/default1/bin/java: No such file or directory
/opt/cloudera/parcels/CDH-5.8.0-1.cdh5.8.0.p0.42/bin/../lib/hadoop-yarn/bin/yarn: line
318:
exec: /usr/java/default1/bin/java: cannot execute: No such file or directory
```

This error occurs when `jdk_home` of the Hadoop distribution is configured incorrectly.

To resolve this error, perform the following steps:

1. Find the `hadoopEnv.properties` in the following directory:
   `<INFA_HOME>/services/shared/hadoop/<Hadoop distribution>/infaConf`
2. Set the `jdk_home` property correctly. For example, you can set the following value:
   ```
   infapdo.env.entry.hadoop_node_jdk_home=HADOOP_NODE_JDK_HOME=/usr/java/default
   ```

I use Sqoop as a Lookup transformation in a streaming mapping. The mapping fails, and I see the following error in the mapping logs:

```
<INFA_HOME>/logs/node_automation/services/DataIntegrationService/disLogs/
ms

:
Caused by: java.io.IOException: Cannot run program "<INFA_HOME>/services/shared/hadoop/
<Hadoop distribution>/scripts/
HadoopFsRmRf" (in directory "."): error=13, Permission denied
at java.lang.ProcessBuilder.start(ProcessBuilder.java:1048)
at java.lang.Runtime.exec(Runtime.java:620)
```

This error occurs when you do not have sufficient permissions on the `<Informatica installation directory>\externaljdbcjars` directory in the Informatica domain. Get the required permissions and then run the mapping again.

For more information about the JDBC driver JAR files for Sqoop connectivity, see the *Informatica Big Data Management Installation and Configuration Guide*.

CHAPTER 6

Window Transformation

This chapter includes the following topics:

Window Transformation Overview

Use the Window transformation when you want to accumulate streamed data into data groups and then process the data sets. The Window transformation is a passive transformation.

When you read from unbounded sources, you might want to accumulate the data into bounded data groups for further processing. To introduce bounded intervals to unbounded data, use a Window transformation.

When you configure a Window transformation, define the type of window and the data boundaries by time. To specify data boundaries, configure the window size and window slide interval. The window size defines the time interval for which data is accumulated as a data group. The slide interval defines the time interval after which the accumulated data group is processed further.

Window Transformation Types

When you create a Window transformation, you can configure sliding or tumbling windows to specify a time interval for selecting a data group from data streams.

Select one of the following window types when you create a Window transformation:

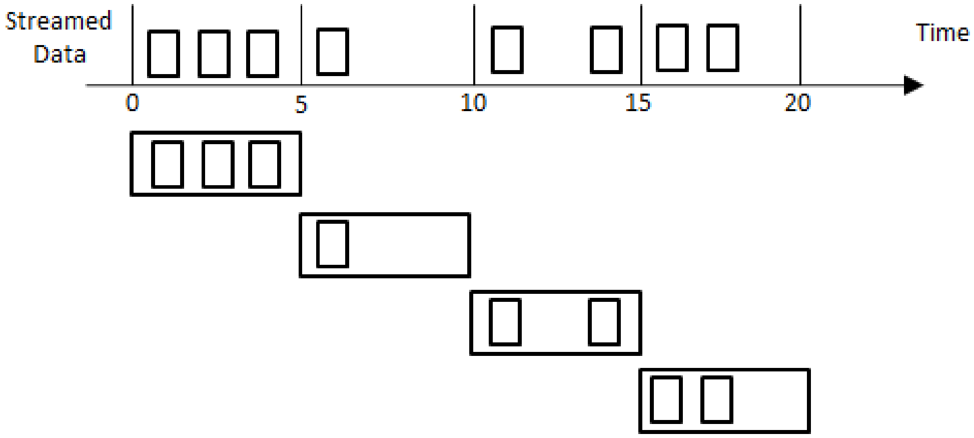
- Tumbling
- Sliding

When you develop a Window transformation, you need to consider factors, such as the type of window, the window size and window slide interval that you want to use on the data that the source streams.

Tumbling Window

A tumbling window accumulates data and returns a bounded data group. After the output data is sent, the tumbling window is cleared and a new group of data is accumulated for the next output. Tumbling windows do not overlap. In tumbling windows, the window size and slide interval are the same.

The following image shows a sample 5-second tumbling window:

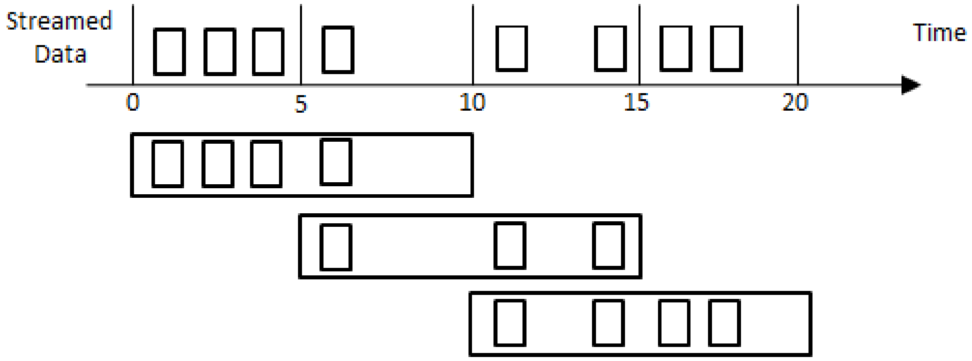


Sliding Window

A sliding window accumulates data and returns a bounded data group. The bounds on the data slide by the time you specify. The data groups that are accumulated can overlap based on the slide interval that you specify.

When you create a sliding Window transformation, the window size must be a multiple of the slide interval.

The following image shows a sample sliding window with a window size of 10 seconds and slide interval of 5 seconds:



Window Transformation Window Properties

A Window transformation has different window types that allow you to accumulate data groups at different time intervals.

Configure the following window properties for a Window transformation:

Window Type

The type of window transformation you want to create. You can choose tumbling or sliding.

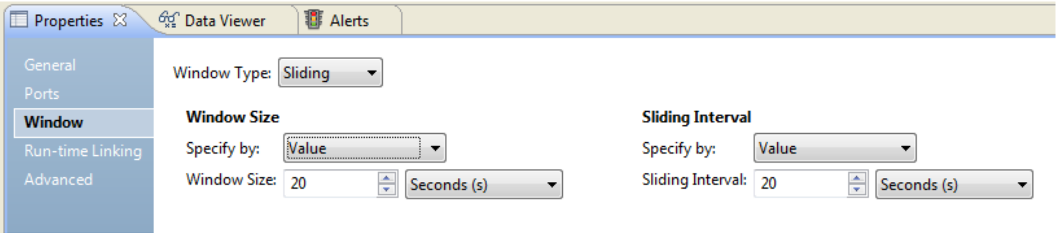
Window Size

The window size defines the time interval for which data is accumulated as a data group. The window size should be a multiple of the batch interval. Specify the window size as a value in units of time or as a parameter of type TimeDuration.

Sliding Interval

The slide interval defines the time interval after which the accumulated data group is processed. Specify the slide interval as a value in units of time or as a parameter of type TimeDuration. Specify the sliding interval if you create a sliding window. By default, the window size and sliding interval are same for tumbling windows.

The following image shows sample window transformation properties:



Window Type: Sliding

Window Size

Specify by: Value

Window Size: 20 Seconds (s)

Sliding Interval

Specify by: Value

Sliding Interval: 20 Seconds (s)

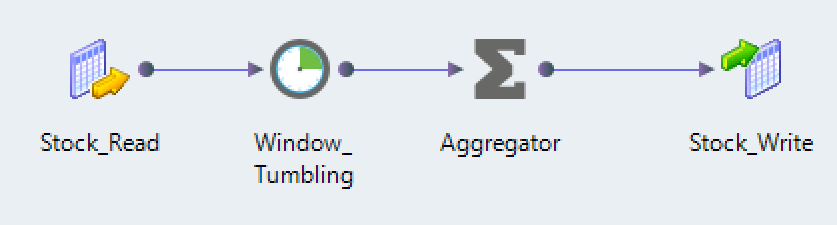
For more information about data types, see the *Informatica Big Data Management User Guide*.

Tumbling Window Transformation Example

You want to calculate the maximum value of a stock price every five minutes for stock prices collected over a five-minute time interval. You can use a tumbling Window transformation.

Create a mapping that reads stock prices and calculates the maximum value every five minute.

The following figure shows the example mapping:



You can use the following objects in your mapping:

Kafka Input

The input, Stock_Read, is a Kafka broker.

Window Transformation

The Window transformation, Window_Tumbling, accumulates data and returns a data group every five minute. Configure a window size of 5 minutes. The default slide interval is 5 minutes. The transformation streams data for five minutes and returns a data group every five minutes.

Aggregator

The Aggregator transformation calculates the maximum value of the stock price.

Kafka Ouptut

The output, Stock_Write, is a Kafka broker.

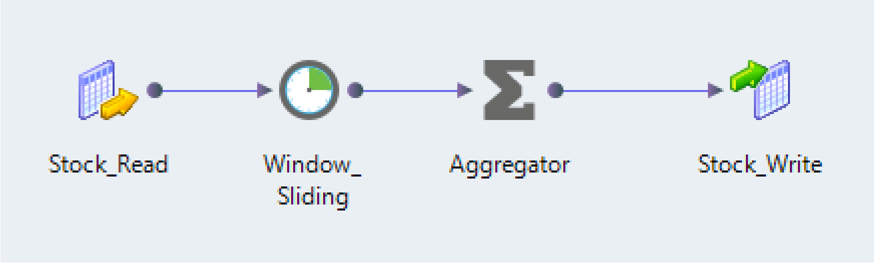
When you run the mapping, the Data Integration Service reads the data from the Kafka broker and passes it to the Window transformation. The window transformation groups the data and provides a data group every five minutes. The Aggregator transformation provides the maximum stock price. The output is written to a Kafka broker.

Sliding Window Transformation Example

You want to calculate the maximum value of a stock price every minute for stock prices collected over a five-minute time interval. You can use a sliding Window transformation.

Create a mapping that reads stock prices and calculates the maximum value every minute.

The following image shows the example mapping:



You can use the following objects in your mapping:

Kafka Input

The input, Stock_Read, is a Kafka broker.

Window Transformation

The Window transformation, Window_Sliding, accumulates data and returns a data group every minute. Configure a window size of 5 minutes and a slide interval of 1 minute. The transformation streams data for five minutes and returns a data group every minute.

Aggregator

The Aggregator transformation calculates the maximum value of the stock price.

Kafka Ouptut

The output, Stock_Write, is a Kafka broker.

When you run the mapping, the Data Integration Service reads the data from the Kafka broker and passes it to the Window transformation. The window transformation groups the data and provides a data group every minute. The Aggregator transformation provides the maximum stock price. The output is written to a Kafka broker.

Rules and Guidelines for Transformations

Certain transformations are valid with restrictions with the Window transformation. The following table describes the rules and guidelines for transformations:

Transformation	Rules and Guidelines
Aggregator	The Aggregator transformation is a multiple-group active transformation. You must use a Window transformation between the streaming source and the Aggregator transformation in a streaming mapping. If you do not specify the group by ports to define groups for aggregations, the transformation sends a value of 0 when it receives no data.
Joiner	The Joiner transformation is a multiple-group active transformation. The following rules apply to Joiner transformations: - You must use a Window transformation between the streaming source and any Joiner transformation in a streaming mapping. - The upstream Window transformations in pipelines to a Joiner transformation must have the same slide intervals.
Lookup	The following rules apply to Lookup transformations: - You can include a Lookup transformation in a streaming mapping if it does not have a streaming source as the input. - You can include a Lookup transformation only if the mapping has flat file or JDBC sources.
Sorter	The Sorter transformation is a multiple-group active transformation. You must use a Window transformation between the streaming source and the Sorter transformation in a streaming mapping.
Union	The following rules apply to Union transformations: - A Union transformation does not require a Window transformation between a streaming source and itself. - If one pipeline to a Union transformation has a Window transformation, all streaming pipelines must have a Window transformation. All upstream Window transformations in the pipelines to the Union transformations must have the same slide interval. - A Union transformation cannot be used to merge data from streaming and non-streaming pipelines.
Window	The following rules apply to Window transformations: - You cannot add a Window transformation to an Logical Data Object mapping, REST mapping, or mapplet. - A Window transformation must have at least one upstream streaming source. - All Window transformations must have a slide interval that is a multiple of the mapping batch interval. - A Window transformation that is downstream from another Window transformation must have a slide interval that is a multiple of the slide interval of the upstream Window transformation. - The slide interval of a sliding Window transformation must be less than window size. - The format of the parameter of the window size must have the TimeDuration parameter type. - The window size and slide interval of a Window transformation must be greater than 0.

INDEX