Informatica

Informatica® Big Data Streaming
10.2.1

Big Data Streaming User Guide

Informatica Big Data Streaming Big Data Streaming User Guide
10.2.1
April 2018

Publication Date: 2022-04-09

Table of Contents

Preface

The Informatica Big Data 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 Big Data Streaming

This chapter includes the following topics:

Big Data Streaming Overview

Use Informatica Big Data Streaming to prepare and process streams of data in real time and uncover insights in time to meet your business needs. Big Data Streaming provides pre-built connectors such as Kafka, Amazon Kinesis, HDFS, enterprise messaging systems, and data transformations to enable a code-free method of defining data integration logic.

Big Data Streaming builds on the best of open source technologies. It uses Spark Streaming for stream processing, and supports other open source stream processing platforms and frameworks, such as Kafka and Hadoop.

Create 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, processes it, and publishes it.

You can create streaming mappings to stream machine, device, and social media data in the form of messages. You can stream data from sources such as JMS providers, Apache Kafka brokers, Amazon Kinesis streams, Microsoft Azure Event Hubs, and MapR streams. Use a Messaging 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, Amazon Kinesis Firehose, HBase tables, Hive tables, JDBC-compliant databases, Microsoft Azure Event Hubs, Azure Data Lake Store, MapR-DB, and MapR streams.

Big Data Streaming works with Informatica Big Data Management to provide streaming capabilities. Big Data 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.

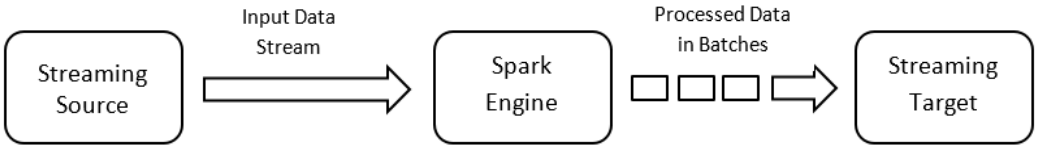
Use Informatica Developer (the Developer tool) to create streaming mappings. Use the Hadoop run-time environment and the Spark engine to run the mapping. You can configure high availability to run the streaming mappings on the Hadoop cluster.

For more information about running mappings on 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 at https://spark.apache.org/documentation.html.

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

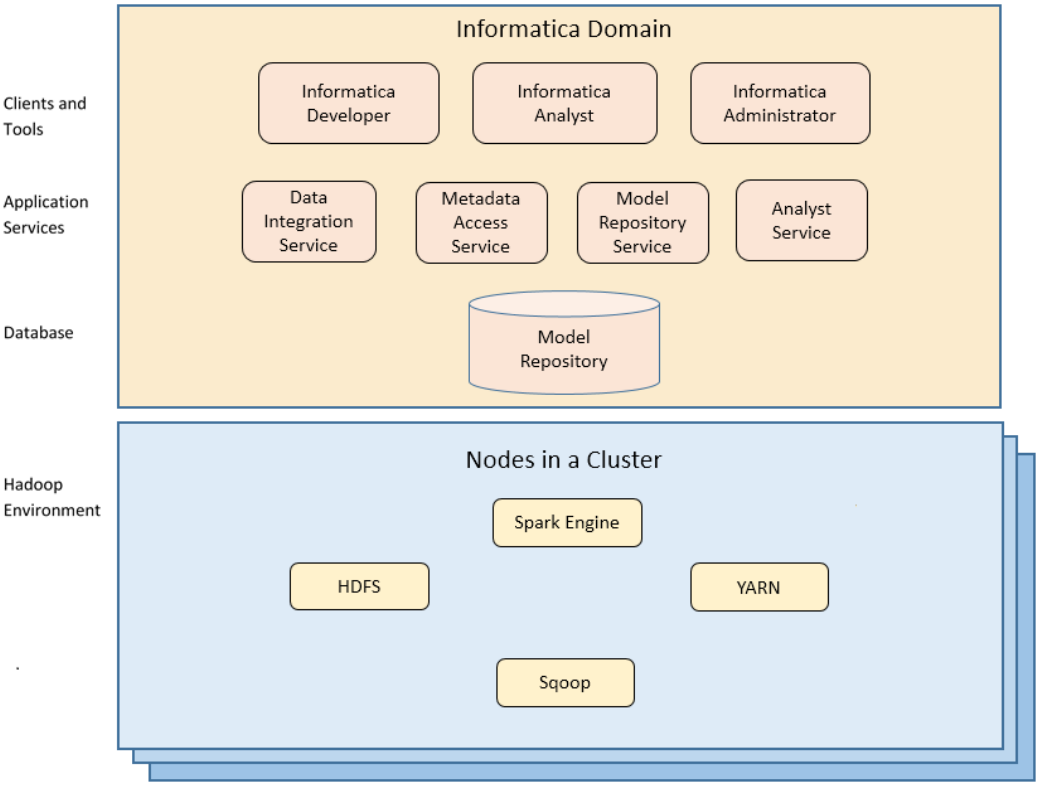
1. Identify sources from which you need to stream data. You can access data that is in XML, JSON, Avro, flat, or binary format.
   You can use Kafka, JMS, Amazon Kinesis stream, Azure Event Hubs, and MapR stream sources to connect to multiple big data sources.
2. Configure the mapping and mapping logic 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, MapR-DB, MapR streams, Azure Event Hubs, Azure Data Lake Store, JMS, Kinesis Firehose delivery streams, 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 Big Data Streaming components for a streaming mapping include client tools, application services, repositories, and third-party tools.

The following image shows the components that Big Data Streaming uses for 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

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

Big Data 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.

Metadata Access Service

The Metadata Access Service is a user-managed service that provides metadata from a Hadoop cluster to the Developer tool at design time. HBase, HDFS, Hive, and MapR-DB connections use the Metadata Access Service when you import an object from a Hadoop cluster. Create and configure a Metadata Access Service before you create HBase, HDFS, Hive, MapR Streams, and MapR-DB connections.

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

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

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

Third-Party Applications

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

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

Example

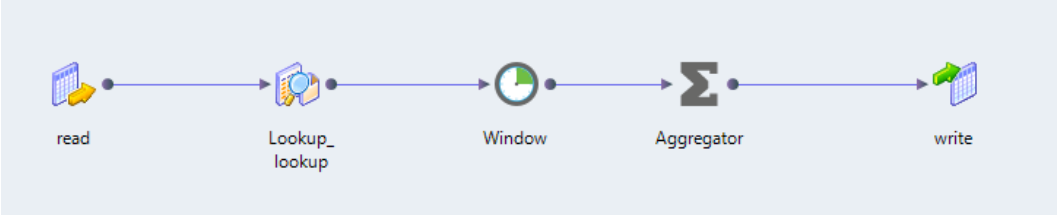
You run the IT department of a major bank that has millions of customers. You want to monitor network activity in real time. You need to collect network activity data from various sources such as firewalls or network devices to improve security and prevent attacks. The network activity data includes 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, CSV, or Avro formats.
2. Add a Lookup transformation to get data from a particular customer ID.
3. Add a Window transformation to accumulate the streamed data into data groups before processing the data.
4. Process the data. Add an Aggregator transformation to perform aggregations on the data from the customer ID.
5. Monitor jobs. Monitor statistics for the mapping job on the Monitoring tab of the Administrator tool.

The following image shows the mapping:



CHAPTER 2

Big Data Streaming Configuration

This chapter includes the following topics:

Big Data Streaming Configuration Overview

After you integrate the Informatica domain with the Hadoop environment, you can use Big Data Streaming in conjunction with Big Data Management.

Before you create streaming mappings in Big Data Streaming, perform the following prerequisite tasks:

- Verify the Hadoop distribution versions that Big Data Streaming supports.
- Before you create and use JMS connections and JMS data objects in streaming mappings, complete the required prerequisites.
- Before you read from or write to a Kerberized Kafka cluster, complete the required prerequisites.
- To configure Sqoop connectivity for relational databases, download JDBC driver jar files.
- To run streaming mappings in a highly available Hadoop cluster, verify the high availability prerequisites.
- If you upgraded the Informatica platform, update the Kafka and AmazonKinesis connection properties.

For more information about integration tasks, see the *Informatica Big Data Management Hadoop Integration Guide*.

Supported Hadoop Distributions

Big Data Streaming supports the following distributions:

- Amazon EMR
- Azure HDInsight with ADLS storage

- Cloudera CDH
- Hortonworks HDP
- MapR

For more information about the distribution versions, see the Product Availability Matrix on Informatica Network:

https://network.informatica.com/community/informatica-network/product-availability-matrices/overview

Prerequisites to Read From or Write to a Kerberised Kafka Cluster

To read from or write to a Kerberised Kafka cluster, configure the default realm, KDC, Hadoop connection properties, and Kafka data object read or write data operation properties.

Before you read from or write to a Kerberized Kafka cluster, perform the following tasks:

1. Ensure that you have the krb5.conf file for the Kerberised Kafka server.
2. Configure the default realm and KDC. If the default `/etc/krb5.conf` file is not configured or you want to change the configuration, add the following lines to the `/etc/krb5.conf` file:

```
[libdefaults]
default_realm = <REALM NAME>
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true

[realms]
<REALM NAME> = {
kdc = <Location where KDC is installed>
admin_server = <Location where KDC is installed>
                                  }
 [domain_realm]
.<domain name or hostname> = <KERBEROS DOMAIN NAME>
<domain name or hostname> = <KERBEROS DOMAIN NAME>
```

3. To pass a static JAAS configuration file into the JVM using the `java.security.auth.login.config` property at runtime, perform the following tasks:
   a. Ensure that you have JAAS configuration file.
   For information about creating JAAS configuration and configuring Keytab for Kafka clients, see the Apache Kafka documentation at https://kafka.apache.org/0101/documentation/#security

   For example, the JAAS configuration file can contain the following lines of configuration:

```
//Kafka Client Authentication. Used for client to kafka broker connection
KafkaClient {
com.sun.security.auth.module.Krb5LoginModule required
doNotPrompt=true
useKeyTab=true
storeKey=true
keyTab="<path to keytab  file>/<keytab file name>"
principal="<principal name>"
client=true
};
```

b. Place the JAAS config file and keytab file in the same location on all the nodes of the Hadoop cluster.
Informatica recommends that you place the files in a location that is accessible to all the nodes in the cluster. Example: `/etc` or `/temp`

c. On the **Spark Engine** tab of the Hadoop connection properties, update the `extraJavaOptions` property of the executor and the driver in the **Advanced Properties** property. Click **Edit** and update the properties in the following format:

```
infaspark.executor.extraJavaOptions=-Djava.security.egd=file:/dev/./urandom
-XX:MaxMetaspaceSize=256M -Djavax.security.auth.useSubjectCredsOnly=true
-Djava.security.krb5.conf=/<path to krb5.conf file>/krb5.conf
-Djava.security.auth.login.config=/<path to jAAS config>/
<kafka_client_jaas>.config

infaspark.driver.cluster.mode.extraJavaOptions=-Djava.security.egd=file:/dev/./
urandom
-XX:MaxMetaspaceSize=256M -Djavax.security.auth.useSubjectCredsOnly=true
-Djava.security.krb5.conf=/<path to krb5.conf file>/krb5.conf
-Djava.security.auth.login.config=<path to jaas config>/
<kafka_client_jaas>.config
```

d. Configure the following properties in the data object read or write operation:

- Data object read operation. Configure the **Consumer Configuration Properties** property in the advanced properties.
- Data object write operation. Configure the **Producer Configuration Properties** property in the advanced properties.

Specify the following value:

```
security.protocol=SASL_PLAINTEXT,sasl.kerberos.service.name=kafka,sasl.mechanism=
GSSAPI
```

4. To embed the JAAS configuration in the `sasl.jaas.config` configuration property, perform the following tasks:

1. On the **Spark Engine** tab of the Hadoop connection properties, update the `extraJavaOptions` property of the executor and the driver in the **Advanced Properties** property. Click **Edit** and update the properties in the following format:

```
infaspark.executor.extraJavaOptions = -Djava.security.egd=file:/dev/./urandom
-XX:MaxMetaspaceSize=256M -XX:+UseG1GC -XX:MaxGCPauseMillis=500
-Djava.security.krb5.conf=<path to krb5.conf file>

infaspark.driver.cluster.mode.extraJavaOptions = -Djava.security.egd=file:/dev/./
urandom
-XX:MaxMetaspaceSize=256M -XX:+UseG1GC -XX:MaxGCPauseMillis=500
-Djava.security.krb5.conf=<path to krb5.conf file>
```

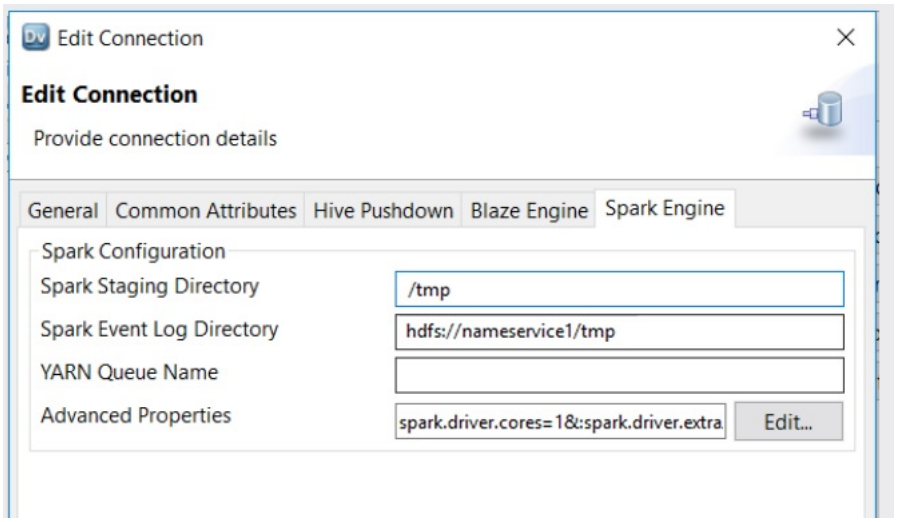
2. Configure the following properties in the data object read or write operation:

- Data object read operation. Configure the **Consumer Configuration Properties** property in the advanced properties.
- Data object write operation. Configure the **Producer Configuration Properties** property in the advanced properties.

Specify the following value:

```
security.protocol=SASL_PLAINTEXT,sasl.kerberos.service.name=kafka,sasl.mechanism=
GSSAPI,
sasl.jaas.config=com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
storeKey=true doNotPrompt=true serviceName="<service_name>" keyTab="<location of
keytab file>"
client=true principal="<principal_name>";
```

The following image shows the **Advanced Properties** property in the Hadoop connection:



Download the JDBC Drivers for Sqoop Connectivity

Download JDBC drivers to configure Sqoop connectivity for relational databases and to use Sqoop as a Lookup transformation.

To configure Sqoop connectivity, perform the following tasks:

1. Download any Type 4 JDBC driver that the database vendor recommends for Sqoop connectivity.
2. To configure Sqoop connectivity for relational databases, copy the jar files to the following directory on the machine where the Data Integration Service runs: `INFA_HOME\services\shared\hadoop\HADOOP_HOME\extras\spark-auxjars\`
3. To configure Sqoop connectivity for lookup, copy the jar files to the following directory: `<Informatica installation directory>\externaljdbcjars`
4. Recycle the Data Integration Service.

At run time, the Data Integration Service copies the jar files to the Hadoop distribution cache so that the jar files are accessible to all nodes in the cluster.

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

High Availability Prerequisites

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

To run streaming mappings in a highly available Hadoop cluster, verify the following prerequisites:

- The Hadoop cluster must be highly available.

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

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

Update Connection Properties

If you upgraded the Informatica platform, update the Kafka and AmazonKinesis connection properties. Review connections that you created in a previous release to update the values for connection properties.

Based on the data objects in the streaming mapping, update the following connections before you run the mapping:

AmazonKinesis

Update the region before you run a streaming mapping that contains an AmazonKinesis data object. Perform the following steps:

1. Delete the existing AmazonKinesis connection and create a new connection with the same name. Specify the region where the endpoint for your service is available.
2. Update the AmazonKinesis data object properties with the new connection details.

Kafka

Configure the version of the Kafka messaging broker. Select **0.8.x.x-0.10.0.x** or **0.10.1.x-1.0.0**.

CHAPTER 3

Sources in a Streaming Mapping

This chapter includes the following topics:

Sources 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, you can create the following data objects:

AmazonKinesis

A physical data object that represents data in an Amazon Kinesis Stream. Create an AmazonKinesis data object to read from an Amazon Kinesis Stream.

Azure Event Hub

A physical data object that represents data in Microsoft Azure Event Hubs data streaming platform and event ingestion service.

HBase

A physical data object that represents data in an HBase resource. Create an HBase data object with a read operation to perform an uncached lookup on HBase data.

JMS

A physical data object that accesses a JMS server. Create a JMS data object to read from a JMS server.

Kafka

A physical data object that accesses a Kafka broker. Create a Kafka data object to read from a Kafka broker.

MapRStreams

A MapRStreams data object is a physical data object that represents data in a MapR Stream. Create a MapRStreams data object to read from a MapR Stream.

Processing Hierarchical Data in Streaming Mappings

Data objects in a streaming mapping can process hierarchical data through complex data types. If the source contains hierarchical data, you must enable the read data operation to project columns as complex data types.

The following table shows the format and complex data types that sources in a Streaming mapping support:

Format	Data Type	Amazon Kinesis Streams	Azure EventHub	JMS	Kafka	MapRStreams
Avro	Flat	Supported	Supported	Not supported	Supported	Supported
Avro	Hierarchical	Supported	Supported	Not supported	Supported	Supported
JSON	Flat	Supported	Supported	Supported	Supported	Supported
JSON	Hierarchical	Supported	Supported	Supported	Supported	Supported
XML	Flat	Supported	Supported	Supported	Supported	Supported
XML	Hierarchical	Supported	Supported	Supported	Supported	Supported

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

AmazonKinesis Data Objects

An AmazonKinesis data object is a physical data object that represents data in a Amazon Kinesis Data Stream. After you create an AmazonKinesis connection, create an AmazonKinesis data object to read from Amazon Kinesis Data Streams.

Kinesis Data Streams is a real-time data stream processing option that Amazon Kinesis offers within the AWS ecosystem. It is a customizable option for users who want to build custom applications to process and analyze streaming data. You must manually provision enough capacity to meet system needs.

When you configure the AmazonKinesis data object, specify the name of the Amazon Kinesis Data Stream that you read from. After you create the data object, create a read operation to read data from an Amazon Kinesis Data Stream. You can then add the data object read operation as a source in streaming mappings.

When you configure the data operation properties, specify the format in which the data object reads data. When you read from Amazon Kinesis Data Stream sources, you can read data in JSON, XML, Avro, Flat, or binary format. When you specify XML format, you must provide a XSD file. When you specify Avro format,

provide a sample Avro schema in a .avsc file. When you specify JSON or Flat format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

Note: You cannot run a mapping with an AmazonKinesis data object on a MapR distribution.

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

For more information about Kinesis Data Streams, see the Amazon Web Services documentation.

Related Topics:

- "Creating a Data Object" on page 85

AmazonKinesis Data Object Overview Properties

Overview properties include general properties that apply to the AmazonKinesis data object. The Developer tool displays overview properties of the data object in the **Overview** view.

You can configure the following overview properties for AmazonKinesis data objects:

General

You can configure the following general properties for the AmazonKinesis data object:

- Name. Name of the AmazonKinesis data object.
- Description. Description of the AmazonKinesis data object.
- Native Name. Name of the AmazonKinesis data object.
- Path Information. The path of the data object in AmazonKinesis. For example, `/DeliveryStreams/router1`

Column

You can configure the name, native name, data type, precision, scale, and description of the columns in the AmazonKinesis resource.

Advanced

The following are the advanced properties for the AmazonKinesis data object:

- Amazon Resource Name. The Kinesis resource that the AmazonKinesis data object is reading from or writing to.
- Type. The type of delivery stream that the AmazonKinesis data object is reading from or writing to. The delivery stream is either Kinesis Stream or Firehose DeliveryStream
- Number of Shards. Specify the number of shards that the Kinesis Stream is composed of. This property is not applicable for Firehose DeliveryStream.

AmazonKinesis Data Object Read Operation Properties

The Data Integration Service uses read operation properties when it reads data from AmazonKinesis Streams.

General Properties

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

The following table describes the general properties for the AmazonKinesis data object read operation:

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

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 AmazonKinesis stream sources:

Property	Description
Name	The name of the 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.
Detail	The detail of the data type.
Scale	The scale of the data type.
Description	The description of the resource.

Sources Properties

The sources properties list the resources of the Amazon Kinesis data object.

The following table describes the sources property that you can configure for Amazon Kinesis Streams sources:

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

Run-time Properties

The run-time properties include properties that the Data Integration Service uses when reading data from the source at run time.

The run-time property for AmazonKinesis Stream source includes the name of the AmazonKinesis connection.

Advanced Properties

The following table describes the advanced properties for AmazonKinesis Stream sources:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
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 AmazonKinesis Stream.
Degree of Parallelism	The number of processes that run in parallel within a shard. Specify a value that is less than or equal to the number of shards.

Column Projections Properties

The following table describes the columns projection properties that you configure for Amazon Kinesis Stream 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. By default, the data is streamed in binary format. To change the format in which the data is processed, select this option and specify the schema format.
Schema Format	The format in which the source processes data. You can select one of the following formats: - XML - JSON - Avro - Flat
Schema	Specify the XSD schema for the XML format, the sample JSON for the JSON format. Specify a .avsc file for the Avro format or a sample file for the Flat format.

Property	Description
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for sources with hierarchical data. Select this option if the source has hierarchical data. For more information on hierarchical data, see the *Informatica Big Data Management User Guide*.

Configuring Scheme for Flat Files

Configure schema for flat files when you configure column projection properties.

1. On the **Column Projection** tab, enable column projection and select the flat schema format.

   The page displays the column projection properties page.
2. On the column projection properties page, configure the following properties:
   - Sample Metadata File. Select a sample file.
   - Code page. Select the UTF-8 code page.
   - Format. Format in which the source processes data. Default value is Delimited. You cannot change it.
3. Click **Next**.
4. In the delimited format properties page, configure the following properties:

Property	Description
Delimiters	Specify the character that separates entries in the file. Default is a comma (,). You can only specify one delimiter at a time. If you select **Other** and specify a custom delimiter, you can only specify a single-character delimiter.
Text Qualifier	Specify the character used to enclose text that should be treated as one entry. Use a text qualifier to disregard the delimiter character within text. Default is No quotes. You can only specify an escape character of one character.
Preview Options	Specify the escape character. The row delimiter is not applicable as only one row is created at a time.
Maximum rows to preview	Specify the rows of data you want to preview.

5. Click **Next** to preview the flat file data object.

   If required, you can change the column attributes. The data type timestampWithTZ format is not supported.
6. Click **Finish**.

   The data object opens in the editor.

Azure EventHub Data Objects

An Azure EventHub data object is a physical data object that represents an event hub object in Microsoft Azure Event Hubs data streaming platform and event ingestion service. After you create an Azure Eventhub connection, create an Azure Eventhub data object to read from event data from Azure Event Hubs.

Azure Event Hubs is a highly scalable data streaming platform and event ingestion service, that receives and processes events. Event Hubs can process and store events or data produced by distributed software and devices.

When you configure the Azure EventHub data object, specify the name of the event hub that you read from. After you create the data object, create a read operation to read event data from an Event Hub. You can then add the data object read operation as a source in streaming mappings.

When you configure the data operation properties, specify the format in which the Azure EventHub data object reads data. You can specify XML, JSON, Avro, or flat as format. When you specify XML format, you must provide a XSD file. When you specify Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON or Flat format, you must provide a sample file.

Note: You can deploy a streaming mapping that has an Event Hub as a source only on a Cloudera CDH distribution.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

Related Topics:

- "Creating a Data Object" on page 85

Azure EventHub Data Object Overview Properties

Overview properties include general properties that apply to the Azure EventHub data object. The Developer tool displays overview properties of the data object in the **Overview** view.

You can configure the following overview properties for Azure EventHub data objects:

General

You can configure the following general properties for the Azure EventHub data object:

- Name. Name of the Azure EventHub data object.
- Description. Description of the Azure EventHub data object.
- Native Name. Name of the Azure EventHub data object.
- Path Information. The path of the data object in Azure EventHub. For example, `/EventHubs/avroevents`

Column

You can configure the name, native name, data type, precision, scale, and description of the columns in the Azure EventHub resource.

Advanced

The following are the advanced properties for the Azure EventHub data object:

- Location. The location of the Event Hub.
- Partition. The number of partitions that the Event Hub has when you import the data object.
- Date of Creation. The date of creation of the Event Hub.

Azure Eventhub Data Object Read Operation Properties

The Data Integration Service uses read operation properties when it reads data from an Azure Event Hub.

General Properties

The Developer tool displays general properties for Azure Event Hub sources in the **Read** view.

The following table describes the general properties for the Azure EventHub data object read operation:

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

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 Azure EventHub sources:

Property	Description
Name	The name of the 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.
Detail	The detail of the data type.
Scale	The scale of the data type.
Description	The description of the resource.

Sources Properties

The sources properties list the resources of the Azure Eventhub data object.

The following table describes the sources property that you can configure for Azure Event Hub events:

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

Run-time Properties

The run-time properties include properties that the Data Integration Service uses when reading data from the source at run time.

The run-time property for Azure Event Hub event includes the name of the Azure Eventhub connection.

Advanced Properties

The following table describes the advanced properties for Azure Event Hub sources:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
Consumer Group	The name of the Event Hub Consumer Group that you read events from.
Max Rate	The maximum number of events that are consumed in a single batch for each partition.
Shared Access Policy Name	The name of the Event Hub Shared Access Policy. To read from Event Hubs, you must have Listen permission. If you specify a value for this property, it overwrites the value configured in the Azure EventHub connection.
Shared Access Policy Primary Key	The primary key of the Event Hub Shared Access Policy. If you specify a value for this property, it overwrites the value configured in the Azure EventHub connection.
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 Azure Event Hub.
Start Position Offset	The time from which the Azure EventHub data object starts reading events from an Event Hub. You can select one of the following options: - CUSTOM. Read messages from a specific time. - EARLIEST Read the earliest messages available on the event hub.
Custom Start Position Timestamp	A UTC timezone datetime value in ISO8601 format from which the Azure EventHub data object starts reading events from an Event Hub. Specify time in the following format: YYYY-MM-DDThhmmss.sssZ

Property	Description
Partition Count	The number of partitions that the Event Hub has. The source reads only from the number of partitions you specify. If the number of partitions on the Event Hub changes, you must change this value to ensure that event from all partitions are consumed.
Consumer Properties	The Event Hub consumer configuration properties. Specify properties as key-value pairs. For example, key1=value1,key2=value2

Column Projections Properties

The following table describes the columns projection properties that you configure for Azure Event Hub 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. By default, the data is streamed in binary format. To change the format in which the data is processed, select this option and specify the schema format.
Schema Format	The format in which the source processes data. You can select one of the following formats: - XML - JSON - Flat - Avro
Schema	Specify the XSD schema for the XML format, the sample JSON for the JSON format. Specify a .avsc file for the Avro format.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for sources with hierarchical data. Select this option if the source has hierarchical data. For more information on hierarchical data, see the *Informatica Big Data Management User Guide*.

HBase Data Objects

An HBase data object is a physical data object that represents data based on an HBase resource. After you create an HBase connection, create an HBase data object and a read data object operation.

Create a data object read operation to when you want to perform an uncached lookup on HBase data.

HBase Data Object Read Operation Properties

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

The Developer tool displays advanced properties for the HBase data object operation in the Advanced view.

The following table describes the Advanced properties for an HBase data object read operation:

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.

JMS Data Objects

A JMS data object is a physical data object that accesses a JMS server. After you configure a JMS connection, create a JMS data object to read from JMS sources.

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 . 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. You can specify XML, JSON, or Flat as format. When you specify XML format, you must provide an XSD file. When you specify JSON, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

Related Topics:

- "Creating a Data Object" on page 85

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 or JSON 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.
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 Overview 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 queue of the JMS source.
Description	The description of the JMS source.
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 Operation Properties

The Data Integration Service uses read operation 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
Operation Type	Specifies the type of data object operation. This is a read-only property.
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.

Property	Description
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. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.
Schema Format	The format in which the source streams data. You can select one of the following formats: - XML - JSON - Flat
Schema	Specify the XSD schema for the XML format or the sample JSON for the JSON format. For the Flat file format, configure the schema to associate a flat file to the source. When you provide a sample file, the Data Integration Service uses UTF-8 code page when reading the data.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

Configuring a Schema for Flat Files

Configure schema for flat files when you configure column projection properties.

1. On the **Column Projection** tab, enable column projection and select the flat schema format.

   The page displays the column projection properties page.

2. On the column projection properties page, configure the following properties:
   - Sample Metadata File. Select a sample file.
   - Code page. Select the UTF-8 code page.
   - Format. Format in which the source processes data. Default value is Delimited. You cannot change it.
3. Click **Next**.
4. In the delimited format properties page, configure the following properties:

Property	Description
Delimiters	Specify the character that separates entries in the file. Default is a comma (,). You can only specify one delimiter at a time. If you select **Other** and specify a custom delimiter, you can only specify a single-character delimiter.
Text Qualifier	Specify the character used to enclose text that should be treated as one entry. Use a text qualifier to disregard the delimiter character within text. Default is No quotes. You can only specify an escape character of one character.
Preview Options	Specify the escape character. The row delimiter is not applicable as only one row is created at a time.
Maximum rows to preview	Specify the rows of data you want to preview.

5. Click **Next** to preview the flat file data object.

   If required, you can change the column attributes. The data type timestampWithTZ format is not supported.
6. Click **Finish**.

   The data object opens in the editor.

Kafka Data Objects

A Kafka data object is a physical data object that represents data in a Kafka stream. After you configure a Messaging connection, create a Kafka data object to read from Apache Kafka brokers.

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.

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. You can create or import a Kafka data object.

When you configure the Kafka data object, specify the topic name that you read from or write to. You can specify the topic name or use a regular expression for the topic name pattern only when you read from Kafka. To subscribe to multiple topics that match a pattern, you can specify a regular expression. When you run the application on the cluster, the pattern matching is done against topics before the application runs. If you add a topic with a similar pattern when the application is already running, the application will not read from the topic.

After you create a Kafka data object, create a read operation. You can use the Kafka data object read operation as a source in streaming mappings. If you want to configure high availability for the mapping, ensure that the Kafka cluster is highly available. You can also read from a Kerberised Kafka cluster.

When you configure the data operation read properties, you can specify the time from which the Kafka source starts reading Kafka messages from a Kafka topic.

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

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

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

For more information about how to use topic patterns in Kafka data objects, see https://docs.informatica.com/data-engineering/data-engineering-streaming/h2l/1132-how-to-use-topic-patterns-in-kafka-data-objects/abstract.html.

Related Topics:

- "Creating a Data Object" on page 85

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

Column Properties

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

Property	Description
Name	The name of the Kafka data object.
Native Name	The native name of the Kafka data object.
Type	The native data type of the Kafka data object.
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 Kafka data object.
Access Type	The type of access the port or column has.

Kafka Data Object Read Operation Properties

The Data Integration Service uses read operation 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 properties that you can configure for Kafka sources:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
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.

Property	Description
Start Position Offset	The time from which the Kafka source starts reading Kafka messages from a Kafka topic. You can select one of the following options: - Custom. Read messages from a specific time. - Earliest. Read the earliest messages available on the Kafka topic. - Latest. Read messages received by the Kafka topic after the mapping has been deployed. This property is applicable for Kafka versions 0.10.1.0 and later.
Custom Start Position Timestamp	The time in GMT from which the Kafka source starts reading Kafka messages from a Kafka topic. Specify a time in the following format: `dd-MM-yyyy HH:mm:ss.SSS` The milliseconds are optional. This property is applicable for Kafka versions 0.10.1.0 and above.
Consumer Configuration Properties	The configuration properties for the consumer. If the Kafka data object is reading data from a Kafka cluster that is configured for Kerberos authentication, include the following property: `security.protocol=SASL_PLAINTEXT,sasl.kerberos.service.name=kafka, sasl.mechanism=GSSAPI`

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. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.

Property	Description
Schema Format	The format in which the source streams data. Select one of the following formats: - XML - JSON - Flat - Avro
Schema	Specify the XSD schema for the XML format, a sample file for JSON, or .avsc file for Avro format. For the Flat file format, configure the schema to associate a flat file to the Kafka source. When you provide a sample file, the Data Integration Service uses UTF-8 code page when reading the data.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

Configuring Schema for Flat Files

Configure schema for flat files when you configure column projection properties.

1. On the **Column Projection** tab, enable column projection and select the flat schema format.

   The page displays the column projection properties page.
2. On the column projection properties page, configure the following properties:
   - Sample Metadata File. Select a sample file.
   - Code page. Select the UTF-8 code page.
   - Format. Format in which the source processes data. Default value is Delimited. You cannot change it.
3. Click **Next**.
4. In the delimited format properties page, configure the following properties:

Property	Description
Delimiters	Specify the character that separates entries in the file. Default is a comma (,). You can only specify one delimiter at a time. If you select **Other** and specify a custom delimiter, you can only specify a single-character delimiter.
Text Qualifier	Specify the character used to enclose text that should be treated as one entry. Use a text qualifier to disregard the delimiter character within text. Default is No quotes. You can only specify an escape character of one character.
Preview Options	Specify the escape character. The row delimiter is not applicable as only one row is created at a time.
Maximum rows to preview	Specify the rows of data you want to preview.

5. Click **Next** to preview the flat file data object.

If required, you can change the column attributes. The data type timestampWithTZ format is not supported.

6. Click **Finish**.

   The data object opens in the editor.

MapR Streams Data Objects

A MapR Streams data object is a physical data object that represents data in a MapR Stream. After you create a MapR Streams connection, create a MapR Streams data object to read data from MapR Streams.

Before you create and use MapR Stream data objects in streaming mappings, complete the required prerequisites.

For more information about the prerequisite tasks, see the *Data Engineering Integration Guide*.

When you configure the MapR Streams data object, specify the stream name that you read from in the following format:

```
/pathname:topic name
```

You can specify the stream name or use a regular expression for the stream name pattern only when you read from MapR Streams. The regular expression that you specify applies to the topic name and not the path name. To subscribe to multiple topics that match a pattern, you can specify a regular expression. When you run the application on the cluster, the pattern matching is done against topics before the application runs. If you add a topic with a similar pattern when the application is already running, the application will not read from the topic.

After you create a MapR Streams data object, create a read data object operation. You can then add the data object read operation as a source in streaming mappings.

You can associate the data object with an intelligent structure model and directly parse input from text, CSV, XML, or JSON input files.

When you configure the data operation properties, specify the format in which the MapR Streams data object reads data. You can specify XML, JSON, or Avro as format. When you specify XML format, you must provide a XSD file. When you specify Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON or Flat format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

For more information about how to use topic patterns in MapR Streams data objects, see https://docs.informatica.com/data-engineering/data-engineering-streaming/h2l/1149-how-to-use-topic-patterns-in-maprstreams-data-objects/abstract.html.

Note: If you use a MapR Streams data object in a streaming mapping, you cannot use an Apache Kafka data object in the same mapping.

Related Topics:

- "Creating a Data Object" on page 85

MapRStreams Object Overview Properties

Overview properties include general properties that apply to the MapRStreams data object. The Developer tool displays overview properties in the Overview view.

General

Property	Description
Name	Name of the MapRStreams data object.
Description	Description of the MapRStreams data object.
Native Name	Native name of the MapR Stream.
Path Information	The path of the MapR Stream.

Column Properties

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

Property	Description
Name	The name of the MapRStreams data object.
Native Name	The native name of the MapRStreams data object.
Type	The native data type of the MapRStreams data object.
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 MapRStreams data object.
Access Type	The access type of the port or column.

MapRStreams Data Object Read Operation Properties

The Data Integration Service uses read operation properties when it reads data from MapR Streams.

General Properties

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

The following table describes the general properties for the MapRStreams data object read operation:

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

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 MapR Stream 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.

Sources Properties

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

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

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

Run-time Properties

The run-time property for MapR Stream source includes the name of the MapRStream connection.

Advanced Properties

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

The following table describes the advanced properties for MapR Stream sources:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
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.

Column Projections Properties

The following table describes the columns projection properties that you configure for MapR Stream 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. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.
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, a sample JSON for the JSON format, or .avsc file for the Avro format.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

CHAPTER 4

Targets in a Streaming Mapping

This chapter includes the following topics:

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 target you are writing to, you can create the following data objects:

AmazonKinesis

A physical data object that represents data in an Amazon Kinesis Firehose Delivery Stream. Create an AmazonKinesis data object to write to an Amazon Kinesis Firehose Delivery Stream.

Azure Event Hub

A physical data object that represents data in Microsoft Azure Event Hubs data streaming platform and event ingestion service. Create an Azure Even Hub data object to connect to an Event Hub target.

Complex file

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

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

HBase

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

JMS

A physical data object that accesses a JMS server. Create a JMS data object to write to a JMS server.

Kafka

A physical data object that accesses a Kafka broker. Create a Kafka data object to write to a Kafka broker.

MapRStreams

A MapRStreams data object is a physical data object that represents data in a MapR Stream. Create a MapRStreams data object to write to a MapR Stream.

Microsoft Azure Data Lake

A Microsoft Azure Data Lake Store data object is a physical data object that represents a Microsoft Azure Data Lake Store table. Create an Azure Data Lake object to write to a Microsoft Azure Data Lake Store table.

Relational

A physical data object that you can use to access a relational table. You can create a relational object to connect to a Hive or JDBC-compliant database.

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

Processing Hierarchical Data in Streaming Mappings

Data objects in a streaming mapping can process hierarchical data through complex data types. If you want to project data as complex data types, you must enable the write data operation to project data as complex data types.

The following table shows the format and complex data types that targets in a Streaming mapping support:

Format	Data Type	Amazon Kinesis Firehose	Azure Data Lake Store	Azure Event Hub	Complex File	JMS	Kafka	MapRStreams
Avro	Flat	Not supported	Supported	Supported	Supported	Not supported	Supported	Supported
Avro	Hierarchical	Not supported	Supported	Supported	Supported	Not supported	Supported	Supported
JSON	Flat	Supported	Supported	Supported	Supported	Supported	Supported	Supported
JSON	Hierarchical	Supported	Supported	Supported	Supported	Supported	Supported	Supported
XML	Flat	Not supported	Supported	Supported	Supported	Supported	Supported	Supported

Format	Data Type	Amazon Kinesis Firehose	Azure Data Lake Store	Azure Event Hub	Complex File	JMS	Kafka	MapRStreams
XML	Hierarchical	Not supported	Supported	Supported	Supported	Supported	Supported	Supported

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

AmazonKinesis Data Objects

An AmazonKinesis data object is a physical data object that represents data in an Amazon Kinesis Data Firehose Delivery Stream. After you create an AmazonKinesis connection, create an AmazonKinesis data object to write to Amazon Kinesis Data Firehose.

Kinesis Data Firehose is a real-time data stream processing option that Amazon Kinesis offers within the AWS ecosystem. Kinesis Data Firehose allows batching, encrypting, and compressing of data. Kinesis Data Firehose can automatically scale to meet system needs.

When you configure the AmazonKinesis data object, specify the name of the Data Firehose Delivery Stream that you write to. You can specify the Kinesis Data Firehose Delivery Stream name or use a regular expression for the stream name pattern. If the input has multiple partitions, you can create multiple Kinesis Data Firehose Delivery Streams to the same target and send the data from these partitions to the individual delivery streams based on the pattern you specify in the stream name.

After you create the data object, create a data object write operation to write data to an Amazon Kinesis Data Firehose Delivery Stream. You can then add the data object write operation as a target in Streaming mappings.

When you configure the data operation properties, specify the format in which the data object writes data. When you write to Amazon Data Firehose targets, you can specify JSON or binary as the format.

When you specify JSON format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

When you run a mapping to write data to an Amazon Kinesis Data Firehose Delivery Stream, the data object uses the AWS Firehose SDK to write data.

Note: You cannot run a mapping with an AmazoznKinesis data object on a MapR distribution.

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

For more information about Kinesis Data Firehose, see the Amazon Web Services documentation.

AmazonKinesis Data Object Overview Properties

Overview properties include general properties that apply to the AmazonKinesis data object. The Developer tool displays overview properties of the data object in the **Overview** view.

You can configure the following overview properties for AmazonKinesis data objects:

General

You can configure the following general properties for the AmazonKinesis data object:

- Name. Name of the AmazonKinesis data object.
- Description. Description of the AmazonKinesis data object.
- Native Name. Name of the AmazonKinesis data object.
- Path Information. The path of the data object in AmazonKinesis. For example, `/DeliveryStreams/router1`

Column

You can configure the name, native name, data type, precision, scale, and description of the columns in the AmazonKinesis resource.

Advanced

The following are the advanced properties for the AmazonKinesis data object:

- Amazon Resource Name. The Kinesis resource that the AmazonKinesis data object is reading from or writing to.
- Type. The type of delivery stream that the AmazonKinesis data object is reading from or writing to. The delivery stream is either Kinesis Stream or Firehose DeliveryStream
- Number of Shards. Specify the number of shards that the Kinesis Stream is composed of. This property is not applicable for Firehose DeliveryStream.

AmazonKinesis Data Object Write Operation Properties

The Data Integration Service uses write operation properties when it writes data to Amazon Kinesis Firehose.

General Properties

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

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

Property	Description
Name	The name of the Amazon Simple Storage Service (Amazon S3), Amazon Redshift tables, or Amazon Elasticsearch Service (Amazon ES) . This property is read-only.
Description	The description of the 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 AmazonKinesis targets:

Property	Description
Name	The 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.
Detail	The detail of the data type.
Scale	The scale of the data type.
Description	The description of the target.

Target Properties

The targets properties list the targets of the Amazon Kinesis data object.

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

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

Run-time Properties

The run-time properties include properties that the Data Integration Service uses when writing data to the target at run time, such as reject file names and directories.

The run-time property for AmazonKinesis targets includes the name of the AmazonKinesis connection.

Advanced Properties

Advanced properties include tracing level, row order, and retry attempt properties.

The following table describes the advanced properties for AmazonKinesis targets:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
Record Delimiter	The record delimiter that is inserted into the Kinesis Firehose delivery stream.

Property	Description
Maximum Error Retry Attempts	The number of times that the Data Integration Service attempts to reconnect to the target.
Response Wait Time (milliseconds)	The number of milliseconds that the Data Integration Service waits for a response to send a batch request.
Retry Attempt Delay Time (milliseconds)	The number of milliseconds that the Data Integration Service waits before it retries to send data to the Kinesis Firehose delivery stream.
Runtime Properties	The runtime properties for the connection pool and AWS client configuration. Specify the properties as key -value pairs. For example: `key1=value1,key2=value2`

Column Projection Properties

The following table describes the columns projection properties that you configure for Amazon Kinesis Firehose targets:

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 write data to the target. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.
Schema Format	The format in which data is written to the target. You can select one of the following formats: - XML - JSON - Flat - Avro
Schema	Specify the XSD schema for the XML format, a sample file for JSON, or .avsc file for Avro format. For the Flat file format, configure the schema to associate a flat file to the source.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information on hierarchical data, see the *Informatica Big Data Management User Guide*.

Azure EventHub Data Objects

An Azure EventHub data object is a physical data object that represents data in Microsoft Azure Event Hubs data streaming platform and event ingestion service. After you create an Azure EventHub connection, create an Azure EventHub data object to write to Event hub events.

Azure Event Hubs is a highly scalable data streaming platform and event ingestion service, that receives and processes events. Event Hubs can process and store events or data produced by distributed software and devices.

When you configure the Azure EventHub data object, specify the name of the event that you write to. After you create the data object, create a data object write operation to write data to an Event Hub. You can then add the data object write operation as a target in streaming mappings.

Configure partition keys to separate events into different partitions. Events with the same partition key are sent to the same partition on the event hub. Events that are sent in a batch do not need to have the same partition key.

When you configure the data operation properties, specify the format in which the Azure EventHub data object writes data. You can specify XML, JSON, Avro, or Flat as format. When you specify XML format, you must provide a XSD file. When you specify Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON or Flat format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

Azure EventHub Data Object Overview Properties

Overview properties include general properties that apply to the Azure EventHub data object. The Developer tool displays overview properties of the data object in the **Overview** view.

You can configure the following overview properties for Azure EventHub data objects:

General

You can configure the following general properties for the Azure EventHub data object:

- Name. Name of the Azure EventHub data object.
- Description. Description of the Azure EventHub data object.
- Native Name. Name of the Azure EventHub data object.
- Path Information. The path of the data object in Azure EventHub. For example, `/EventHubs/avroevents`

Column

You can configure the name, native name, data type, precision, scale, and description of the columns in the Azure EventHub resource.

Advanced

The following are the advanced properties for the Azure EventHub data object:

- Location. The location of the Event Hub.
- Partition. The number of partitions that the Event Hub has when you import the data object.
- Date of Creation. The date of creation of the Event Hub.

Azure EventHub Data Object Write Operations

The Data Integration Service uses write operation properties when it writes data to an Azure Event Hub.

General Properties

The Developer tool displays general properties for Azure Event Hub sources in the **Write** view.

The following table describes the general properties for the Azure EventHub data object write operation:

Property	Description
Name	The name of the Azure EventHub data object. This property is read-only. You can edit the name in the **Overview** view. When you use the Azure Event Hub as a target in a mapping, you can edit the name in the mapping.
Description	The description of the Azure EventHub data object operation.

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 Azure EventHub targets:

Property	Description
Name	The 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.
Detail	The detail of the data type.
Scale	The scale of the data type.
Description	The description of the resource.

Target Properties

The targets properties list the targets of the Azure EventHub data object.

The following table describes the sources property that you can configure for Azure Event Hub targets:

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

Run-time Properties

The run-time properties include properties that the Data Integration Service uses when writing data to the target at run time, such as reject file names and directories.

The run-time property for Azure Event Hub targets includes the name of the Azure EventHub connection.

Advanced Properties

The following table describes the advanced properties for Azure Event Hub targets:

Property	Description
Operation Type	Specifies the type of data object operation. This is a read-only property.
Batch Size	The batch size of the events. Specify an integer value for the batch size.
Shared Access Policy Name	The name of the Event Hub Namespace Shared Access Policy. To write to Event Hubs, you must have Send permissions. If you specify a value for this property, it overwrites the value configured in the Azure EventHub connection.
Shared Access Policy Primary Key	The primary key of the Event Hub Namespace Shared Access Policy. If you specify a value for this property, it overwrites the value configured in the Azure EventHub connection.
Publisher Properties	The Event Hub publisher configuration properties. Specify properties as key-value pairs. For example, key1=value1,key2=value2

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 Azure EventHub 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. By default, the data is streamed in binary format. To change the streaming format, select this option and specify the schema format.
Schema Format	The format in which you stream data to the target. You can select one of the following formats: - XML - JSON - Flat - Avro
Schema	Specify the XSD schema for the XML format or the sample file for JSON or Avro format. For the Flat file format, configure the schema to associate a flat file.
Column Mapping	The mapping of data object to the target. Click **View** to see the mapping.
Project as Hierarchical Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

Complex File Data Objects

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

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 format in which the complex file data object writes data to the HDFS sequence file. You can also specify binary as format.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

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.
Native Name	Name of the HDFS connection.
Path Information	The path on the Hadoop file system.

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.

Compression Options	Description
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 Operation Properties

The Data Integration Service uses write operation 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. 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.
Scale	The scale for each column. Scale is the maximum number of digits that a column can accommodate to the right of the decimal point. Applies to decimal columns. The scale values you configure depend on the data type.

Property	Description
Detail	The detail of the data object.
Description	The description of the resource.

Target Properties

The target 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.

Run-time Properties

The run-time properties include the name of the connection that the Data Integration Service uses to write data to the HDFS sequence file or binary file.

You can configure dynamic partitioning or fixed partitioning.

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
Operation Type	Indicates the type of data object operation. This is a read-only property.
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 one of the following file formats: - Binary. Select Binary to read any file format. - Sequence. Select Sequence File Format for target files of a specific format that contain key and value pairs.

Property	Description
Output Format	The class name for files of the output format. If you select Output Format in the **File Format** field, you must specify the fully qualified class name implementing the `OutputFormat` interface.
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. By default, the columns are projected as binary data type To change the format in which the data is projected, select this option and specify the schema format.
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.

Property	Description
Column Mapping	Click **View** to see the mapping of the data object to target mapping.
Project as Hierarchical Type	Project columns as complex data type for hierarchical data. For more information on hierarchical data, see the *Informatica Big Data Management User Guide*.

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 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 columns 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.

When you manually create the data object, the column name should be of the following format:
`columnFamily__columnQualifier`

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 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 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.
Native Name	Native name of the HBase data object.
Path	Path to the HBase data object.

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.
Precision	Precision of the data.
Scale	Scale of the data.

HBase Data Object Write Operation Properties

The Data Integration Service uses write operation 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
Operation Type	The type of data object operation. This is a read-only property.
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

A JMS data object is a physical data object that accesses a JMS server. After you configure a JMS connection, create a JMS data object to write to JMS targets.

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

The JMS data object can 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 write data operation properties, specify the format in which the JMS data object writes data. You can specify XML, JSON, or Flat as format. When you specify XML format, you must provide an XSD file. When you specify JSON or Flat format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

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 or JSON 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.
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 Overview 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 queue of the JMS source.
Description	The description of the JMS source.
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 Write Operation Properties

The Data Integration Service uses write operation 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
Operation Type	Specifies the type of data object operation. This is a read-only property.
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.
Message Type	The format in which the message is written to the target. Specify one of the following formats: - Text - Binary

Column Projections Properties

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

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 publish the data to the target. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.
Schema Format	The format in which you stream data to the target. You can select one of the following formats: - XML - JSON - Flat
Schema	Specify the XSD schema for the XML format, and the sample JSON for the JSON format. For the Flat file format, configure the schema to associate a flat file.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.

Kafka Data Objects

A Kafka data object is a physical data object that represents data in a Kafka stream. After you configure a Messaging connection, create a Kafka data object to write to Apache Kafka brokers.

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 write data to a Kafka messaging stream, specify the name of the topic that you publish to. You can also write to a Kerberised Kafka cluster.

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.

After you create a Kafka data object, create a write operation. You can use 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 writes data. You can specify XML, JSON, Avro, or Flat as format. When you specify XML format, you must provide a XSD file. When you specify Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON or Flat format, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

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

Column Properties

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

Property	Description
Name	The name of the Kafka data object.
Native Name	The native name of the Kafka data object.
Type	The native data type of the Kafka data object.
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 Kafka data object.
Access Type	The type of access the port or column has.

Kafka Data Object Write Operation Properties

The Data Integration Service uses write operation 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
Operation Type	Specifies the type of data object operation. This is a read-only property.
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.
Producer Configuration Properties	The configuration properties for the producer. If the Kafka data object is writing data to a Kafka cluster that is configured for Kerberos authentication, include the following property: `security.protocol=SASL_PLAINTEXT,sasl.kerberos.service.name=kafka, sasl.mechanism=GSSAPI`

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. By default, the data is streamed in binary format. To change the streaming format, select this option and specify the schema format.
Schema Format	The format in which you stream data to the target. You can select one of the following formats: - XML - JSON - Flat - Avro

Property	Description
Schema	Specify the XSD schema for the XML format, acsample file for JSON, or .avsc file for Avro format. For the Flat file format, configure the schema to associate a flat file to the Kafka target. When you provide a sample file, the Data Integration Service uses UTF-8 code page when writing the data.
Column Mapping	The mapping of data object to the target. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

Microsoft Azure Data Lake Store Data Object

A Microsoft Azure Data Lake Store data object is a physical data object that represents data in a Microsoft Azure Data Lake Store table. After you create an Azure Data Lake Store connection, create a Microsoft Azure Data Lake store data object write operation to write to a Microsoft Azure Data Lake Store table.

You can use Microsoft Azure Data Lake Store to store data irrespective of size, structure, and format. Use Microsoft Azure Data Lake Store to process large volumes of data to achieve faster business outcomes.

When you configure the data operation properties, specify the format in which the 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 Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON, you must provide a sample file.

You cannot run a mapping with a Microsoft Data Lake Store data object on a MapR distribution.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

Microsoft Azure Data Lake Store Data Object Properties

The Microsoft Azure Data Lake Store Overview view displays general information about the Microsoft Azure Data Lake Store data object and the object properties that apply to the Microsoft Azure Data Lake Store table you import.

General Properties

You can configure the following properties for a Microsoft Azure Data Lake Store data object:

- Name. Name of the Microsoft Azure Data Lake Store data object.
- Description. Description of the Microsoft Azure Data Lake Store data object.
- Connection. Name of the Microsoft Azure Data Lake Store connection.

Microsoft Azure Data Lake Store Data Object Write Operation Properties

The Data Integration Service writes data to a Microsoft Azure Data Lake Store object based on the data object write operation. The Developer tool displays the data object write operation properties for the Microsoft Azure Data Lake Store data object in the Data Object Operation section.

You can view the data object write operation from the Input and Target properties.

Input properties

Represent data that the Data Integration Service reads from a Microsoft Azure Data Lake Store directory server. Select the input properties to edit the port properties and specify the advanced properties of the data object write operation.

Target properties

Represent data that the Data Integration Service writes to Microsoft Azure Data Lake Store. Select the target properties to view data, such as the name and description of the Microsoft Azure Data Lake Store object.

Input Properties of the Data Object Write Operation

Input properties represent data that the Data Integration Service writes to a Microsoft Azure Data Lake Store directory server. Select the input properties to edit the port properties of the data object write operation. You can also specify advanced data object write operation properties to write data to Microsoft Azure Data Lake Store objects.

The input properties of the data object write operation include general properties that apply to the data object write operation. Input properties also include port, source, and advanced properties that apply to the data object write operation.

You can view and change the input properties of the data object write operation from the **General**, **Ports**, **Targets**, **run-time**, and **Advanced** tabs.

Ports Properties - Input Write

The input ports properties list the data types, precision, and scale of the data object write operation.

The following table describes the input ports properties that you must configure in the data object write operation:

Property	Description
Name	Name of the port.
Type	Data type of the port.
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 port.

Run-time Properties

The run-time properties display the name of the connection used for write transformation.

The following table describes the run-time properties that you configure for a Microsoft Azure Data Lake Store write operation:

Property	Description
Connection	Name of the Microsoft Azure Data Lake Store connection.

Advanced Properties

You can use the advanced properties to specify data object write operation properties to write data to a Microsoft Azure Data Lake Store server.

The following table describes the advanced properties that you configure for a Microsoft Azure Data Lake Store write operation:

Property	Description
Tracing Level	By default, the tracing level for every transformation is Normal. Change the tracing level to a Verbose setting when you need to troubleshoot a transformation that is not behaving as expected. Set the tracing level to Terse when you want the minimum amount of detail to appear in the log.
Maintain row order	NA
If file exists	The streaming mapping ignores this value. A new target file is created.

Column Projection 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 Azure Data Lake Store 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. By default, the data is streamed in binary format. To change the streaming format, select this option and specify the schema format.

Property	Description
Schema Format	The format in which you stream data to the target. You can select one of the following formats: - XML - JSON - Avro
Schema	Specify the XSD schema for the XML format, asample file for JSON, or .avsc file for Avro format.
Column Mapping	The mapping of data object to the target. Click **View** to see the mapping.
Project as Hierarchical Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

MapRStreams Data Objects

A MapRStreams data object is a physical data object that represents data in a MapR Stream. After you create a MapRStreams connection, create a MapRStreams data object to write data to MapR Streams.

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

For more information about the prerequisite tasks, see the *Informatica Big Data Management Cluster Integration Guide*.

When you write data to a MapR stream, specify the name of the stream that you publish to.

After you create a MapRStreams data object, create a write data object operation. You can then add the data object write operation as a target in Streaming mappings.

When you configure the data operation properties, specify the format in which the MapR Streams 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 Avro format, provide a sample Avro schema in a .avsc file. When you specify JSON, you must provide a sample file.

You can pass any payload format directly from source to target in Streaming mappings. You can project columns in binary format pass a payload from source to target in its original form or to pass a payload format that is not supported.

Streaming mappings can read, process, and write hierarchical data. You can use array, struct, and map complex data types to process the hierarchical data. You assign complex data types to ports in a mapping to flow hierarchical data. Ports that flow hierarchical data are called complex ports.

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

MapRStreams Object Overview Properties

Overview properties include general properties that apply to the MapRStreams data object. The Developer tool displays overview properties in the Overview view.

General

Property	Description
Name	Name of the MapRStreams data object.
Description	Description of the MapRStreams data object.
Native Name	Native name of the MapR Stream.
Path Information	The path of the MapR Stream.

Column Properties

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

Property	Description
Name	The name of the MapRStreams data object.
Native Name	The native name of the MapRStreams data object.
Type	The native data type of the MapRStreams data object.
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 MapRStreams data object.
Access Type	The access type of the port or column.

MapRStreams Data Object Write Operation Properties

The Data Integration Service uses write operation properties when it writes data to MapR Streams.

General Properties

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

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

Property	Description
Name	The name of the MapR Stream target. This property is read-only.
Description	The description of the MapR Stream target.

Port 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 MapR Stream targets:

Property	Description
Name	The 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.

Target Properties

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

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

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

Run-time Properties

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

The run-time property for MapR Stream target includes the name of the MapRStream connection.

Column Projection Properties

The following table describes the columns projection properties that you configure for MapR Stream targets:

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 write data to the target. By default, the data is streamed in binary format. To change the format in which the data is streamed, select this option and specify the schema format.
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 .avsc file for the Avro format.
Column Mapping	The mapping of source data to the data object. Click **View** to see the mapping.
Project Column as Complex Data Type	Project columns as complex data type for hierarchical data. For more information, see the *Informatica Big Data Management User Guide*.

Relational Data Objects

Create a relational data object to write to Hive tables or JDBC-compliant database. To write to Hive tables, create a relational data object with a Hive connection. To write to a JDBC-compliant database, create a relational data object with a JDBC connection.

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.

Hive targets

When you write to a Hive target in a streaming mapping, you write to a Hive table. You can write to the following types of tables:

- Managed or internal tables. When you write to a managed table or an internal table, Hive writes data to the Hive warehouse directory. 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.
- External tables. When you write to an external table, you must truncate the target table to overwrite data. You can write to external partitioned tables but you cannot truncate external partitioned tables.

Truncation of tables happens only once in the beginning when you write data.

JDBC targets

You can include a JDBC-compliant database as a target in an Informatica mapping. Use the Sqoop arguments in the JDBC connection to configure the JDBC parameters. Sqoop uses the values that you configure in the **User Name** and **Password** fields of the JDBC connection. If you configure the --username or --password argument in a JDBC connection or mapping, Sqoop ignores the arguments. If you create a password file to access a database, Sqoop ignores the password file.

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.

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.
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.

Property	Description
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 Operation Properties

The Data Integration Service uses write operation properties to write data to Hive or JDBC-compliant database.

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

General Properties

The general properties 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. 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.

Data Object Properties

On the Data Object tab, you can specify or change the target, and make relational and customized data object targets dynamic.

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

Property	Description
Specify By	To specify target columns and metadata, select one of the following options: - Value. The write operation uses the associated data object to specify target columns and metadata. - Parameter. The write operation uses a parameter to specify target columns and metadata. Default is the Value option.
Data Object	If you created the target from an existing data object, the field displays the name of the object. Click Browse to change the data object to associate with the data object.
Parameter	Choose or create a parameter to associate with the target.
At runtime, get data object columns from data source	When you enable this option, the Data Integration Service fetches metadata and data definition changes from target tables to the data object.

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 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.

Advanced Properties

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

Property	Description
Tracing level	Controls the amount of detail in the mapping log file.
Create or replace table at run time	The Data Integration Service drops the target table at run time and replaces it with a table based on a target table that you identify. **Note:** When you write to a JDBC target and the target does not have a table, Spark creates a table even if you do not select this option. Spark creates columns in the table with appropriate target data type and default precision for the data type irrespective of the the precision you specify in the target properties.
DDL Query	The DDL query based on which the Data Integration Service creates or replace the target table at run time. You cannot create a custom DDL query that creates or replaces a Hive table at run time in streaming mappings.
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.
Truncate Hive Target Partition	Truncates the external table before loading data. Streaming mappings do not support this property. The property is ignored.
PreSQL	The SQL command the Data Integration Service runs against the target database before it reads the source. The Developer tool does not validate the SQL.

Property	Description
PostSQL	The SQL command that the Data Integration Service runs against the target database after it writes to the target. The Developer tool does not validate the SQL. Streaming mappings do not support Post-SQL queries. Post-SQL queries in streaming mappings are ignored.
Maintain Row Order	Maintain row order while writing data to the target. Select this option if the Data Integration Service should not perform any optimization that can change the row order. When the Data Integration Service performs optimizations, it might lose the row order that was established earlier in the mapping. You can establish row order in a mapping with a sorted flat file source, a sorted relational source, or a Sorter transformation. When you configure a target to maintain row order, the Data Integration Service does not perform optimizations for the target.

CHAPTER 5

Streaming Mappings

This chapter includes the following topics:

Streaming Mappings Overview

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

Develop a mapping to read, transform, and write data according to your business needs. 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.

Use the following steps as a guideline when you develop a streaming mapping:

1. Create connections that you want to use to access streaming data.
2. Create input, output, and reusable objects that you want to use in the mapping. Create physical data objects to use as mapping input or output.
3. Create reusable transformations that you want to use.
4. Create rules.
5. Create the streaming mapping.
6. Add objects to the mapping.

7. Link ports between mapping objects to create a flow of data from sources to targets, through transformations.
8. Configure the mapping
9. Validate the mapping to identify errors.
10. Save the mapping and run it to see the mapping output.

Connections

A connection is a repository object that defines a connection in the domain configuration repository.

Create a connection to import data objects, preview data, and run mappings.

For more information about the connections that you can use in streaming mappings, see the Appendix, "Connections", in this guide.

Data Objects

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

The following table lists the data objects that you can include in streaming mappings and the read and write data object operations for each:

Data Object	Source	Data Object Operation	Target	Data Object Operation
AmazonKinesis	Amazon Kinesis Stream	Read	Amazon Kinesis Firehose Delivery Stream	Write
Azure Data Lake Store	-	-	Microsoft Azure Data Lake Store	Write
Azure Event Hub	Azure Event Hub Service	Read	Azure Event Hub Service	Write
Complex File	-	-	HDFS sequence file or binary file	Write
HBase	-	-	HBase tables	Write
JMS	JMS server	Read	JMS server	Write
Kafka	Kafka broker	Read	Kafka broker	Write
MapRStreams	MapR Stream	Read	MapR Stream	Write
Relational	-	-	Hive table JDBC-compliant database	Write

Related Topics:

- "Complex File Data Objects" on page 54
- "HBase Data Objects" on page 59
- "JMS Data Objects" on page 62
- "Kafka Data Objects" on page 67
- "Microsoft Azure Data Lake Store Data Object" on page 71
- "MapRStreams Data Objects" on page 74
- "Relational Data Objects" on page 77
- "AmazonKinesis Data Objects" on page 21
- "Azure EventHub Data Objects" on page 26
- "JMS Data Objects" on page 30
- "Kafka Data Objects" on page 35
- "MapR Streams Data Objects" on page 41

Creating a Data Object

Create a data object to add to a streaming mapping.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File** > **New** > **Data Object**.
3. Select the data object that you want to add t the streaming mapping and click **Next**.

   The data object dialog box appears.
4. Enter a name for the data object.
5. Click **Browse** next to the **Location** option and select the target project or folder.
6. Click **Browse** next to the **Connection** option and select the connection that you want to use
7. To add a resource, click **Add** next to the **Selected Resources** option.

   The **Add Resource** dialog box appears.
8. Select the check box next to the resource you want to add and click **OK**.
9. Click **Finish**.
   The data object appears under Data Objects in the project or folder in the **Object Explorer** view.

Related Topics:

- "AmazonKinesis Data Objects" on page 21
- "Azure EventHub Data Objects" on page 26
- "JMS Data Objects" on page 30
- "Kafka Data Objects" on page 35
- "MapR Streams Data Objects" on page 41
- "Complex File Data Objects" on page 54
- "HBase Data Objects" on page 59
- "JMS Data Objects" on page 62
- "Kafka Data Objects" on page 67
- "Microsoft Azure Data Lake Store Data Object" on page 71

- "MapRStreams Data Objects" on page 74
- "Relational Data Objects" on page 77

Creating a Data Object Operation

You can create the data object read or write operation for a data object depending on the source that you read from or target that you write to. You can then add the object operation to a streaming mapping.

1. Select the data object in the **Object Explorer** view.
2. Right-click and select **New** > **Data Object Operation**.

   The **Data Object Operation** dialog box appears.
3. Enter a name for the data object operation.
4. Select the type of data object operation. You can choose to create a StreamRead or StreamWrite operation.

   The Data Integration Service uses read operation properties when it reads data and write operation properties when it writes data.
5. Click **Add**.

   The **Select Resources** dialog box appears.
6. Select the data object for which you want to create the data object operation and click **OK**.
7. Click **Finish**.

The Developer tool creates the data object operation for the selected data object.

Related Topics:

- "AmazonKinesis Data Object Read Operation Properties" on page 22
- "AmazonKinesis Data Object Write Operation Properties" on page 48
- "Azure Eventhub Data Object Read Operation Properties" on page 27
- "Azure EventHub Data Object Write Operations" on page 52
- "JMS Data Object Read Operation Properties" on page 32
- "JMS Data Object Write Operation Properties" on page 64
- "Kafka Data Object Read Operation Properties" on page 37
- "Kafka Data Object Write Operation Properties" on page 68
- "MapRStreams Data Object Read Operation Properties" on page 42
- "MapRStreams Data Object Write Operation Properties" on page 75
- "Complex File Data Object Write Operation Properties" on page 56
- "HBase Data Object Write Operation Properties" on page 62
- "Microsoft Azure Data Lake Store Data Object Write Operation Properties" on page 72
- "Relational Data Object Write Operation Properties" on page 79

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.

You can use the following transformations in a streaming mapping:

Aggregator

Mapping validation fails when the

- The transformation contains stateful variable ports.
- The transformation contains unsupported functions in an expression.

Data Masking

Supported without restrictions.

Expression

Mapping validation fails when 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.

To run user code directly on the Spark engine, the JDK version that the Data Integration Service uses must be compatible with the JRE version on the cluster. For best performance, create the environment variable DIS_JDK_HOME on the Data Integration Service in the Administrator tool. The environment variable contains the path to the JDK installation folder on the machine running the Data Integration Service. For example, you might enter a value such as `/usr/java/default`

You can use complex data types to process hierarchical data.

Note: The complex data type support for the Java transformation is available for technical preview. Technical preview functionality is supported but is unwarranted and is not production-ready. Informatica recommends that you use in non-production environments only.

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

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 and perform an uncached lookup on HBase data.

Mapping validation fails in the following situations:

- Case sensitivity is disabled.
- The lookup is a data object.

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.

When you configure the transformation to return the first, last, or any value on multiple matches, the Data Integration Service returns any value.

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.

To use a Lookup transformation on uncached HBase tables, perform the following steps:

1. Create an HBase data object. When you add an HBase table as the resource for a HBase data object, include the ROW ID column.
2. Create a HBase read data operation and import it into the streaming mapping.
3. When you import the data operation to the mapping, select the **Lookup** option.
4. In the Lookup tab, configure the following options:
   - Lookup column. Specify an equality condition on ROW ID
   - Operator. Specify =
5. Verify that format for any date value in the HBase tables is of a valid Java date format. Specify this format in the **Date Time Format** property of the **Advanced Properties** tab of the data object read operation.

Mapping validation fails in the following situations:

- If you do not include ROW ID in the condition
- If you specify any operator other than =
- If you include multiple conditions in the transformation.
- If you select column of type date from input columns.
- If you look up binary data.

Normalizer

Supported without restrictions.

Python

Supported without restrictions.

Note: The Python transformation is available for technical preview. Technical preview functionality is supported but is unwarranted and is not production-ready. Informatica recommends that you use in non-production environments only.

Rank

Mapping validation fails if case-sensitivity is disabled.

Router

Supported without restrictions.

Sorter

To use the Sorter transformation in a Streaming mapping, configure the following properties:

- Advanced properties of the data object write properties. Enable the **Maintain Row Order** field.
- Custom properties of the Data Integration Service. Set the `ExecutionContextOptions.Infa.HonorTargetOrdering` property to true if there are one or more transformations between the Sorter transformation and the target.

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 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 that are not listed here are not supported.

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

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

Stateful Computing

Stateful computing involves storing and retrieving state while evaluating expressions in an Expression transformation.

You can use variable port definitions in Expression transformations to perform stateful computing in a streaming mapping.

Use variable port definitions to perform the following tasks:

- Calculate and store the state in stateful variables when an event arrives.
- Use the stored state and input to compute the output.

When you configure the Expression transformation, configure the partition keys on the **Windowing** tab. When you define variable ports in an Expression transformation, you can optionally specify a partition key that uses one or more input ports. The partition keys determine which columns to group the data by while performing stateful computing. The stored state will have one value for every stateful variable for each value of the partition key.

The evaluation of the ports is ordered. The output ports are computed after the variable ports are computed and contain updated values of variables.

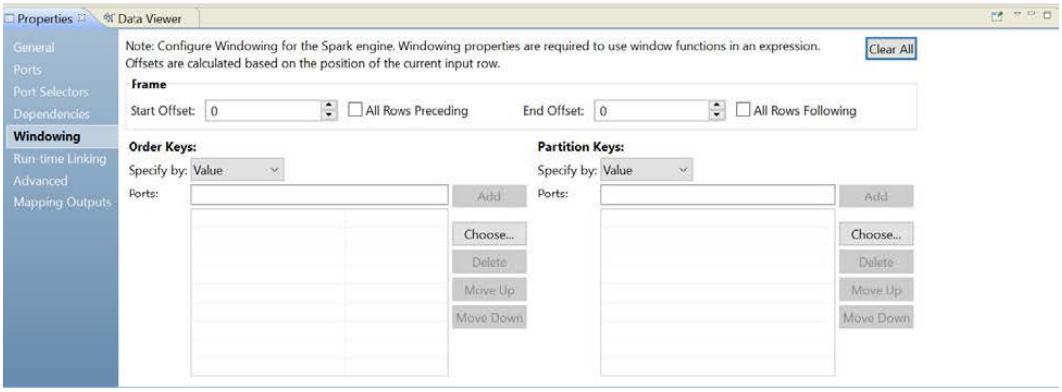
When you configure windowing properties, you define a stateful expression in the Expression transformation. Streaming mappings support all the expression transform function except Window functions and aggregate functions.

For more information about guidelines for configuring variable ports, see the *Informatica Developer Transformation Guide*.

Partitioning Configuration

Configure partitioning by specifying one or more columns under partition keys on the **Windowing** tab.

The following image shows the Windowing tab:



Optionally, configure the partition keys to separate the input rows into different partitions. Configure the partition keys to define partition boundaries, rather than performing the calculation across all inputs. If you do not define partition keys, all the data is included in the same partition. The variable values stored will be global and every row can read and update the same set of variables.

You can specify the partition keys by value or parameter. Select **Value** to use port names.

The following table lists the data type support for variable ports:

Data Type	Initial State Value
String	EMPTY_STRING
Integer	0
Double	0.0

Data Type	Initial State Value
Long	0
Text	EMPTY_STRING
Decimal	0.0

Example

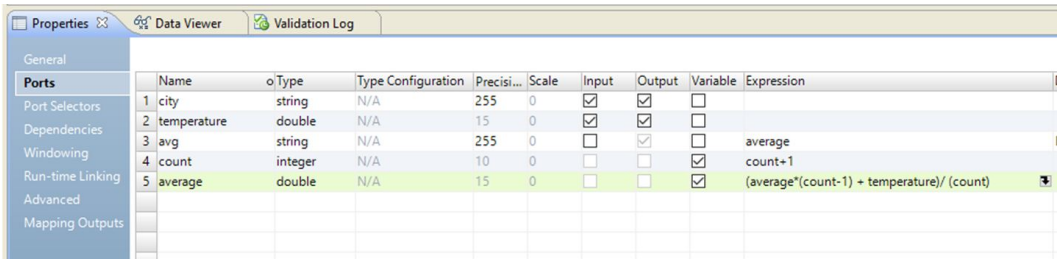
You want to compute average temperature of cities in a region. Add an Expression transformation to your streaming mapping.

Configure the following ports in the transformation:

- city. Input port of string data type.
- temperature. Input port of double data type.
- avg. Output port that displays the average temperature.
- count. Variable port of integer data type with expression ‘count+1’.
- average. Variable port of double data type with expression (average * (count-1) + temperature/(count)

You partition the data by "city". The "average" corresponds to previously stored value for the "average" and is computed with each update in the value of "count". Since "count" is already incremented when "average" is evaluated, specify "count-1" in the expression to get the new average.

The following image shows an Expression transformation:



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 Configurations

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

When you configure the mapping, configure the following 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 Properties. 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 or Amazon Kinesis Streams 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
  ```

In the case of a mapping failure, to enable the mapping to start reading data from the time of failure, configure the `infaspark.checkpoint.directory` property. For example: `infaspark.checkpoint.directory <directory>`. The directory you specify is created under the directory you specify in the **State Store** property.

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.

Run Configurations

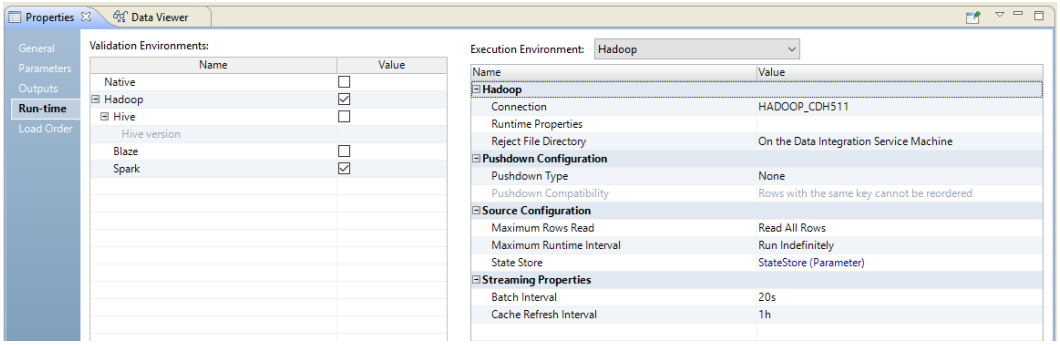
The Developer tool applies configuration properties when you run streaming mappings. Set configuration properties for streaming mappings in the **Run** dialog box.

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.

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.

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



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.

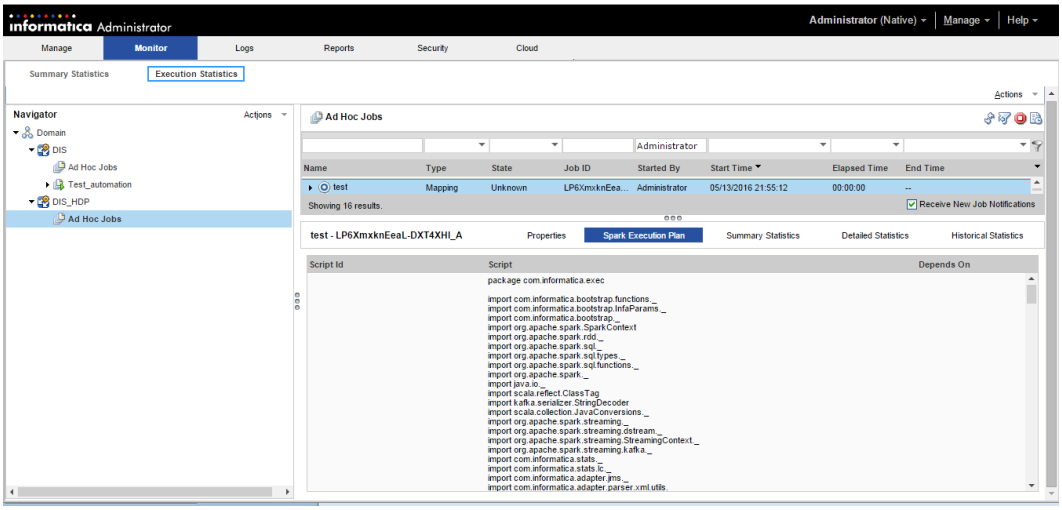
The Developer tool generates an error in the following scenarios:

- 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:



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.

Streaming Mapping 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.

Create a Streaming mapping to read the network activity data and write the data to HDFS.

You can use the following objects in the Streaming mapping:

Kafka data object

The input file is a Kafka queue that contains the network activity data.

Create a Kafka data object. Configure a Kafka connection and specify the queue that contains the network activity data as a resource for the data object. Create a data object read operation and configure the properties. Drag the data object into the mapping as a source data object.

Transformations

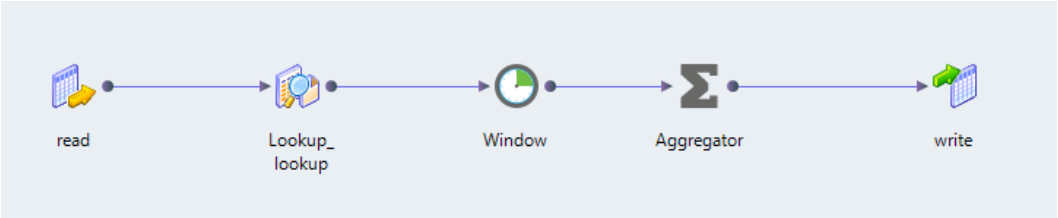
Add a Lookup transformation to get data from a particular customer ID. Add a Window transformation to accumulate the streamed data into data groups before processing the data.

HDFS complex file data object

Create a complex file data object. Configure an HDFS connection to write to an HDFS sequence file. Create the data object write operation and configure the properties. Drag the data object into the mapping as a target data object.

Link ports between mapping objects to create a flow of data.

The following image shows the sample mapping:



When you run the mapping, the data is read from the Kafka queue and written to the HDFS sequence file.

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 cluster environment variables.

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. Edit the Hadoop connection in the Developer tool or the Administrator tool.
2. On the Common Attributes tab, edit the **Cluster Environment Variables** property
3. Set the `HADOOP_NODE_JDK_HOME` property correctly.

When I run a Streaming mapping that contains an HBase data object, I get the following error:

```
HBaseDataAdapter : java.lang.NullPointerException
at
com.informatica.products.extensions.adapter.hadoop.hive.storagehandler.utils.PwxWriter.cl
ose(PwxWriter.java:165)
at
com.informatica.products.extensions.adapter.hadoop.hive.storagehandler.PwxHiveRecordWrite
r.close(PwxHiveRecordWriter.java:119)
at com.informatica.platform.dtm.executor.hive.boot.storagehandler.INFAOutputFormat
$INFAHiveRecordWriter.close(INFAOutputFormat.java:145)
at org.apache.spark.sql.hive.SparkHiveWriterContainer.close(hiveWriterContainers.scala:
109)
at
org.apache.spark.sql.hive.SparkHiveWriterContainer.writeToFile(hiveWriterContainers.scala
:194)
at org.apache.spark.sql.hive.execution.InsertIntoHiveTable$$anonfun$saveAsHiveFile
$3.apply(InsertIntoHiveTable.scala:131)
at org.apache.spark.sql.hive.execution.InsertIntoHiveTable$$anonfun$saveAsHiveFile
$3.apply(InsertIntoHiveTable.scala:131)
at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:70)
at org.apache.spark.scheduler.Task.run(Task.scala:86)
at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:274)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor
```

This error occurs when you try to write a null value to a ROW column of an HBase table.

Ensure that you do not write a null value to a ROW column of an HBase table.

When I test a MapRStreams connection, the Developer tool crashes.

This error occurs if you have not completed the required prerequisites.

Ensure that you copy the conf files to the following directory: `<INFA_HOME>\clients\DeveloperClient\hadoop\mapr_5.2.0\conf`

For more information about the prerequisite tasks, see the *Informatica Big Data Management Cluster Integration Guide*.

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

1. Edit the Hadoop connection in the Developer tool or the Administrator tool.

2. On the Common Attributes tab, edit the **Cluster Environment Variables** property
3. Set the `HADOOP_NODE_JDK_HOME` property correctly.

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 Cluster Integration Guide*.

When I import a data object with Avro schema in a streaming mapping the mapping fails with the following error:

```
com.informatica.adapter.sdkadapter.exceptions.AdapterSDKException: [SDK_APP_COM_20000]
error [getSchemaConfig()::java.io.IOException: Not a data file.]
```

This occurs when the sample schema file is invalid. When you specify Avro format, provide a valid Avro schema in a .avsc file.

When I run a streaming mapping with an Event Hub source or target, the mapping fails with the following error:

```
com.microsoft.azure.eventhubs.AuthorizationFailedException: Put token failed. status-
code: 401, status-description: InvalidSignature: The token has an invalid signature.
```

This occurs when the Shared Access Policy Name or Shared Access Policy Primary Key is configured incorrectly in the Azure EventHub read data operation, Azure EventHub write data operation, or Azure Eventhub connection. Configure valid values for these properties. When you configure the properties in the Azure EventHub connection, ensure that the policy applies to all data objects that are associated with the connection.

A streaming mapping created in 10.2.0 might not run after upgrading to 10.2.1 when you use JSON schema for column projection for the data you are writing

This might occur if the payload has malformed data.

Remove the malformed data run the mapping.

When I run a streaming mapping on a Cloudera CDH cluster that contains a Kafka source or target, the mapping fails.

This error occurs because the default value of the `Offset Commit Topic Replication Factor` property for the Kafka broker is 3 and you are running the mapping on a Cloudera CDH cluster with one or two nodes.

Perform the following steps to resolve this error:

1. Delete all Kafka topics.

2. Stop the Kafka broker and clear the Kafka log directory specified in the `log.dirs` property of the Kafka broker.
3. Stop Zookeeper and clear the log directory specified in the `dataDir` property of ZooKeeper.
4. Configure the `Offset Commit Topic Replication Factor` property for the Kafka broker. Specify a value of 1 or 2 for `offsets.topic.replication.factor` property depending on the number of nodes in the cluster.
5. Start ZooKeeper and the Kafka broker.
6. Create the Kafka topics and run the mapping again.

When I run a streaming mapping that contains a JMS target, I get the following error:

```
WebSphere MQ call failed with compcode '2' ('MQCC_FAILED') reason
'2053' ('MQRC_Q_FULL').
at com.ibm.msg.client.wmq.common.internal.Reason.createException
```

This error occurs when the JMS queue that you are writing to is full.

Increase the queue depth of the JMS server and run the mapping again.

In a streaming mapping, when I perform a self-join on source data that has metadata of complex data type, the mapping validation fails at design time.

This error might occur if you have selected the **Sorted Input** property in the advanced properties of the Joiner transformation.

To resolve this error, deselect the **Sorted Input** property and run the mapping again.

When I run a streaming mapping that contains a JMS source, it fails with an `Unexpected JMSMessage payload type` error.

This error might occur in the following situations:

- There is a mismatch between the data type that you write to the queue and the data present in the queue. Clear the JMS queue and then run the mapping.
- There is a mismatch in the data type that you configured for the JMS source and the data type of the streamed data. Verify that the data type that you configure for the source is the same as the data type of the streamed data.

When I edit the schema of a data that is of complex data type, the schema type does not change.

This error occurs because the **Project Column as Complex Data Type** option is not selected.

To resolve this error, when you edit the schema, select the **Project Column as Complex Data Type** option in the columns projection properties of the data object read or write operation properties.

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:

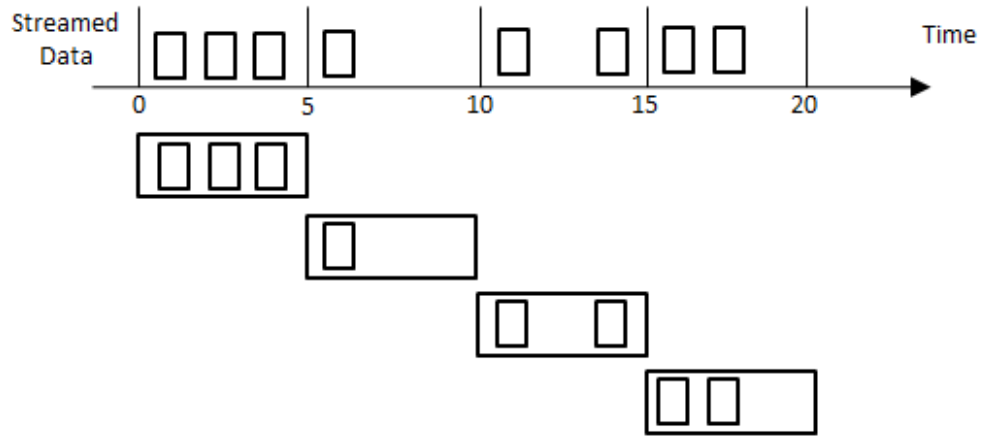
- Tumbling
- Sliding

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:

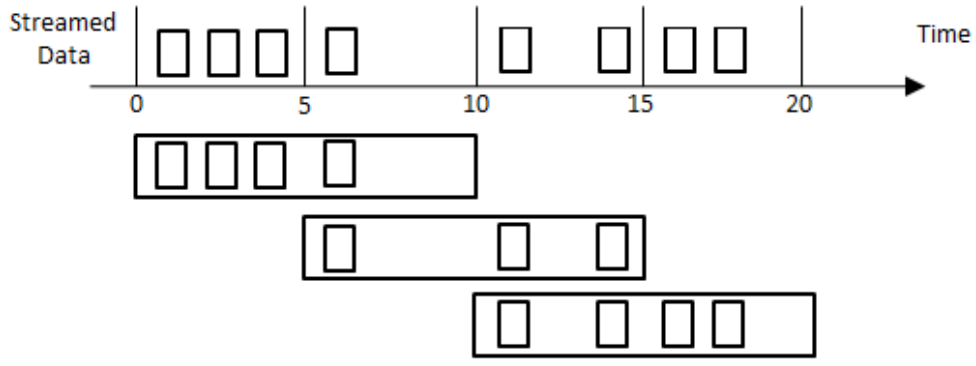


Sliding Window

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.

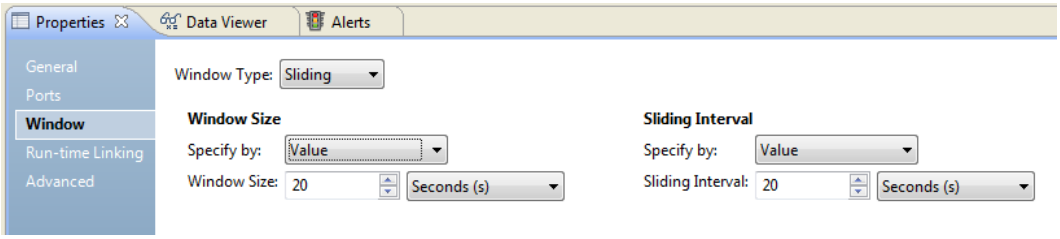
Window Size

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:



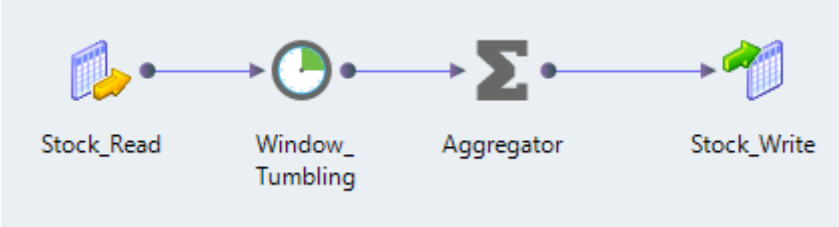
For more information about data types, see the *Informatica Big Data Management User Guide*.

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.

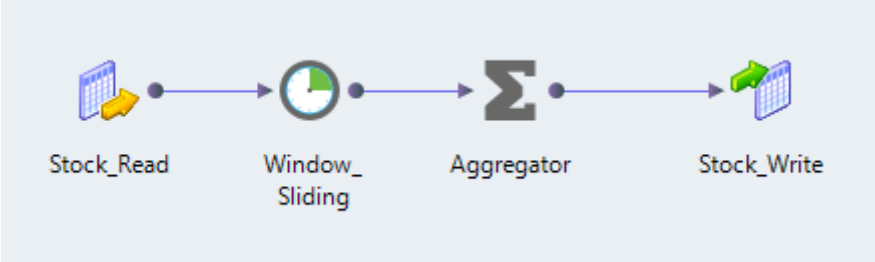
When you run the mapping, the Data Integration Service reads the data from the Kafka broker and passes it to the Window transformation. The window transformation groups the data and provides a data group every five minutes. The Aggregator transformation provides the maximum stock price. The output is 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 before an 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: - A Lookup transformation does not require a Window transformation between a streaming source and itself. - You can include a Lookup transformation only if the mapping has flat file, Hive, HBase, or JDBC sources.
Rank	You must use a Window transformation before a Rank transformation in a Streaming mapping.
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.

APPENDIX A

Connections

This appendix includes the following topics:

Connections Overview

Define the connections that you want to use to access data in Kafka brokers, JMS servers, HDFS files, Hive tables, Amazon Kinesis streams, MapR streams 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 binary or sequence file.

Hive

Create a Hive connection to write data to Hive tables.

For more information, see the *Informatica Big Data Management Administrator Guide*.

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*.

Microsoft Azure Data Lake Store

Create a Microsoft Azure Data Lake Store connection to write to a Microsoft Azure Data Lake Store.

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:

- AmazonKinesis. Create an AmazonKinesis connection to read from Amazon Kinesis Streams or write to Amazon Kinesis Firehose Delivery Streams.
- AzureEventHub. Create an AzureEventHub connection to read from or write to Microsoft Event Hubs.
- JMS. Create a JMS connection to read from or write to a JMS server.
- Kafka. Create a Kafka connection to read from or write to a Kafka broker.
- MapRStreams. Create a MapRStreams connection read from or write to MapR Streams.

AmazonKinesis Connection

The AmazonKinesis connection is a Messaging connection. Use the AmazonKinesis connection to access Amazon Kinesis Data Streams as source or Amazon Kinesis Data Firehose as target. You can create and manage an AmazonKinesis connection in the Developer tool or through infacmd.

When you configure an AmazonKinesis connection, you configure the following properties:

- Type of service.
- Access keys (access key ID and secret access key) to interact with Amazon Web Services.
- Region that hosts the service that you are trying to use.

Prerequisites

Before you create an AmazonKinesis connection, perform the following tasks:

1. Generate an Access Key ID and Secret Access Key for the user in AWS. You can provide these key values when you create an AmazonKinesis connection.
   If you do not provide these key values, the values are picked up from the AWS configuration file location on the Developer client machine, during metadata import. Similarly, at runtime, the values are picked up from the default AWS configuration file location on the Hadoop cluster.

2. If you use Amazon Kinesis Streams as a source, perform the following task:
    a. Verify that the credentials that belong IAM user that you specify in the Access Key ID has the consumer permissions that are part of the IAM policy.
    For the list of permissions, see the AWS documentation at https://docs.aws.amazon.com/streams/latest/dev/learning-kinesis-module-one-iam.html
    b. Verify that you have the following permissions to fetch metadata: kinesis:DescribeStream, kinesis:GetShardIterator, and kinesis:GetRecords
3. If you use the Amazon Kinesis Firehose service as a target, perform the following tasks:
    a. Verify that you have an AWS account with the required IAM permissions to use the AWS services Kinesis Firehose, S3, Redshift, and Elastic Search.
    b. If you are writing to Amazon Redshift, verify that you have the Redshift INSERT privilege to copy data from the Amazon S3 bucket to the Redshift cluster.
    c. Define a Firehose Delivery Stream with either S3 or Redshift or Elasticsearch as its destination. Configure source as Direct PUT or other sources.
    d. Verify that the IAM user credentials have required permissions, based on the target you are writing to.
    For a list of permissions, see the AWS documentation at https://docs.aws.amazon.com/firehose/latest/dev/controlling-access.html#access-to-firehose

General Properties

The following table describes the general connection properties for the AmazonKinesis 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/AmazonKinesis.

Connection Properties

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

Property	Description
Service	The type of Kinesis Service that the connection is associated with. Select one of the following service types: - Kinesis Firehose. Select this service to write to Kinesis Firehose Delivery Stream. - Kinesis Streams. Select this service to read from Kinesis Streams
AWS Access Key ID	The access key ID of the Amazon AWS user account.
AWS Secret Access Key	The secret access key for your Amazon AWS user account.
Region	Region where the endpoint for your service is available. You can select one of the following values: - us-east-2. Indicates the US East (Ohio) region. - us-east-1. Indicates the US East (N. Virginia) region. - us-west-1. Indicates the US West (N. California) region. - us-west-2. Indicates the US West (Oregon) region. - ap-northeast-1. Indicates the Asia Pacific (Tokyo) region. - ap-northeast-2. Indicates the Asia Pacific (Seoul) region. - ap-northeast-3. Indicates the Asia Pacific (Osaka-Local) region. - ap-south-1. Indicates the Asia Pacific (Mumbai) region. - ap-southeast-1. Indicates the Asia Pacific (Singapore) region. - ap-southeast-2. Indicates the Asia Pacific (Sydney) region. - ca-central-1. Indicates the Canada (Central) region. - cn-north-1. Indicates the China (Beijing) region. - cn-northwest-1. Indicates the China (Ningxia) region. - eu-central-1. Indicates the EU (Frankfurt) region. - eu-west-1. Indicates the EU (Ireland) region. - eu-west-2. Indicates the EU (London) region. - eu-west-3. Indicates the EU (Paris) region. - sa-east-1. Indicates the South America (São Paulo) region.
Connection Timeout (in msec)	Number of milliseconds that the Integration service waits to establish a connection to the Kinesis Stream or Kinesis Firehose after which it times out.

Creating an AmazonKinesis Connection using Informatica Command Line Program

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

On UNIX, run the following command:

```
sh infacmd.sh createConnection
```

On Windows, run the following command:

```
infacmd.bat createConnection
```

Enter connection options in the following format:

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

For example, to create an AmazonKinesis connection to Kinesis Streams on UNIX, run the following command:

```
infacmd createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct AMAZONKINESIS -o AWS_ACCESS_KEY_ID=<access key id>
AWS_SECRET_ACCESS_KEY=<secret access key> ConnectionTimeOut=10000 Region=<RegionName>
ServiceType='Kinesis Streams'
```

To create an AmazonKinesis connection to Kinesis Firehose on UNIX, run the following command:

```
infacmd createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct AMAZONKINESIS -o AWS_ACCESS_KEY_ID=<access key>
AWS_SECRET_ACCESS_KEY=<secret access key> ConnectionTimeOut=10000 Region=<RegionName>
ServiceType='Kinesis Firehose'
```

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

Azure EventHub Connection

Use the Azure EventHub connection to access Azure Event Hub as source or target. You can create and manage an Azure EventHub connection in the Developer tool or through infacmd.

Prerequisites

Before you create an Azure Eventhub connection, verify the following prerequisites:

1. You must have a Microsoft Azure account with a minimum role of contributor.
2. Verify that you have an Active Directory application created.
3. Verify that your Active Directory application has permissions for the following API:
   - Windows Azure Service Management API
   - Windows Azure Active Directory
4. Verify that the Azure Active Directory application is added with a reader role to your Azure account subscription.
5. Verify that you have an EventHub Namespace.

General Properties

The following table describes the general connection properties for the AzureEventHub 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 Messaging/AzureEventHub

Connection Properties

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

Property	Description
Tenant ID	The ID of the tenant that the data belongs to. This ID is the Directory ID of the Azure Active Directory.
Subscription ID	The ID of the Azure subscription.
Resource Group Name	The name of the Azure resource group associated with the event hub namespace.
Client Application ID	The ID of the application created under the Azure Active Directory.
Client Secret Key	The secret key generated for the application.
Event Hub Namespace	The name of the Event Hub Namespace that is associated with the resource group name.
Shared Access Policy Name	The name of the Event Hub Namespace Shared Access Policy. The policy must apply to all data objects that are associated with this connection. To read from Event Hubs, you must have Listen permission. To write to an Event hub, the policy must have Send permission.
Shared Access Policy Primary Key	The primary key of the Event Hub Namespace Shared Access Policy.

Creating an Azure EventHub Connection using Informatica Command Line Program

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

On UNIX, run the following command:

```
sh infacmd.sh createConnection
```

On Windows, run the following command:

```
infacmd.bat 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 Azure EventHub connection:

```
sh infacmd.sh createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct AZUREEVENTHUB -o tenantID='tenant id'
subscriptionID='subscription id' resourceName='resource name' clientID='client id'
clientSecretKey='secret key' eventHubNamespace='namespace' sasPolicyName=<policy name>
sasPolicyPrimaryKey=<policy key>
```

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 on the folloiwng options: - HBase - MapR-DB

Connection Properties

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

Property	Description
Database Type	The connection type. Select on the following options: - HBase - MapR-DB
Cluster Configuration	The name of the cluster configuration associated with the Hadoop environment. For more information about cluster configuration, see the *Informatica Big Data Management Administrator Guide*.
MapR-DB Database Path	Database path that contains the MapR-DB table that you want to connect to. Enter a valid MapR cluster path. When you create an HBase data object for MapR-DB, you can browse only tables that exist in the MapR-DB path that you specify in the Database Path field. You cannot access tables that are available in sub-directories in the specified path. For example, if you specify the path as /user/customers/, you can access the tables in the customers directory. However, if the customers directory contains a sub-directory named regions, you cannot access the tables in the following directory: `/user/customers/regions`

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 createConnection
```

On Windows, run the following command:

```
infacmd.bat 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:

```
sh infacmd.sh createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct HBase -o DatabaseType=HBase CLUSTERCONFIGID=ConfId
```

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 binary or 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.
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 the `maprfs:///` format to specify the NameNode URI in MapR distribution.
Cluster Configuration	The name of the cluster configuration associated with the Hadoop environment. For more information about cluster configuration, see the *Informatica Big Data Management Administrator Guide*.

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 createConnection
```

On Windows, run the following command:

```
infacmd.bat 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 createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct HadoopFileSystem -o NameNodeURL='hdfs://
<namenode>:<port>' USERNAME='username' clusterConfigId='ConfId'
```

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.

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 createConnection
```

On Windows, run the following command:

```
infacmd.bat 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 CreateConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -ct HIVE -o "CLUSTERCONFIGID='ConfId'
relationalSourceAndTarget='true' pushDownMode='false' metaDataConnString='jdbc:hive2://
<hostname>:<port>/<db>' enableQuotes='false' HIVEWAREHOUSEDIRECTORYONHDFS='/user/hive/
warehouse' DATABASENAME='default' BYPASSHIVEJDBCSERVER='false' ConnectString='jdbc:hive2://
<hostname>:<port>/<db> databaseName=default defaultFSURI= hdfs://<node name>:<port>'"
```

For more information about the CreateConnection command and Hive connection options, 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.

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 Big Data Streaming.

To create a JMS connections, copy 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. Verify that the IBM MQ client is installed on all the cluster nodes.

2. Copy the JAR files from the IBM MQ server.
3. Place the JAR files in the following directory on the machine where the Data Integration Service runs:

   `INFA_HOME/services/shared/hadoop/<Hadoop distribution>/extras/spark-auxjars/`

   For example, `/CDH_5.13/extras/spark-auxjars`
4. Create the `.bindings` file using the IBM MQ Server.

   The `.bindings` file must have the details of the MQ Objects that the data objects in the streaming mappings use.
5. Copy the `.bindings` file to the Developer tool machine.
6. When you create the JMS connection, configure the **Connection URL** property with the location of the `.bindings` file on the Developer tool machine.
   - On Windows, specify the following format for the location: `file:////<Path to .bindings file>`
     For example: `file:////C:/JMS`
   - On Linux, specify the following format for the location: `file:///<Path to .bindings file>`
7. Place the `.bindings` file in the same path on all the cluster nodes.

   The default location of the files is `/etc`

   For example, `file:///etc`
8. Before you run the mapping, update the **Connection URL** property with the location of the `.bindings` file on the cluster nodes.

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.
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 createConnection
```

On Windows, run the following command:

```
infacmd.bat 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:

```
sh infacmd.sh createConnection -dn <domain name> -un <domain name> -pd <domain password> -cn
<connection name> -cid <connection id> -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*.

JDBC Connection Properties

You can use a JDBC connection to access tables in a database. You can create and manage a JDBC connection in the Administrator tool, the Developer tool, or the Analyst tool.

Note: The order of the connection properties might vary depending on the tool where you view them.

The following table describes JDBC connection properties:

Property	Description
Database Type	The database type.
Name	Name of the connection. The name is not case sensitive and must be unique within the domain. The name cannot exceed 128 characters, contain spaces, or contain the following special characters: `~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /`
ID	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. The description cannot exceed 765 characters.
User Name	The database user name. If you configure Sqoop, Sqoop uses the user name that you configure in this field. If you configure the --username argument in a JDBC connection or mapping, Sqoop ignores the argument.
Password	The password for the database user name. If you configure Sqoop, Sqoop uses the password that you configure in this field. If you configure the --password argument in a JDBC connection or mapping, Sqoop ignores the argument.

JDBC Driver Class Name	Name of the JDBC driver class.
	The following list provides the driver class name that you can enter for the applicable database type:
	- DataDirect JDBC driver class name for Oracle:
	`com.informatica.jdbc.oracle.OracleDriver`
	- DataDirect JDBC driver class name for IBM DB2:
	`com.informatica.jdbc.db2.DB2Driver`
	- DataDirect JDBC driver class name for Microsoft SQL Server:
	`com.informatica.jdbc.sqlserver.SQLServerDriver`
	- DataDirect JDBC driver class name for Sybase ASE:
	`com.informatica.jdbc.sybase.SybaseDriver`
	- DataDirect JDBC driver class name for Informix:
	`com.informatica.jdbc.informix.InformixDriver`
	- DataDirect JDBC driver class name for MySQL:
	`com.informatica.jdbc.mysql.MySQLDriver`
	For more information about which driver class to use with specific databases, see the vendor documentation.

Property	Description

Connection String	Connection string to connect to the database. Use the following connection string:
	`jdbc:<subprotocol>:<subname>`
	The following list provides sample connection strings that you can enter for the applicable database type:
	- Connection string for DataDirect Oracle JDBC driver:
	`jdbc:informatica:oracle://<host>:<port>;SID=<value>`
	- Connection string for Oracle JDBC driver:
	`jdbc:oracle:thin:@//<host>:<port>:<SID>`
	- Connection string for DataDirect IBM DB2 JDBC driver:
	`jdbc:informatica:db2://<host>:<port>;DatabaseName=<value>`
	- Connection string for IBM DB2 JDBC driver:
	`jdbc:db2://<host>:<port>/<database_name>`
	- Connection string for DataDirect Microsoft SQL Server JDBC driver:
	`jdbc:informatica:sqlserver://<host>;DatabaseName=<value>`
	- Connection string for Microsoft SQL Server JDBC driver:
	`jdbc:sqlserver://<host>;DatabaseName=<value>`
	- Connection string for Netezza JDBC driver:
	`jdbc:netezza://<host>:<port>/<database_name>`
	- Connection string for Pivotal Greenplum driver:
	`jdbc:pivotal:greenplum://<host>:<port>;/database_name=<value>`
	- Connection string for Postgres Greenplum driver:
	`jdbc:postgressql://<host>:<port>/<database_name>`
	- Connection string for Teradata JDBC driver:
	`jdbc:teradata://<host>/database_name=<value>,tmode=<value>,charset=<value>`
	For more information about the connection string to use with specific drivers, see the vendor documentation.

Environment SQL	Optional. Enter SQL commands to set the database environment when you connect to the database. The Data Integration Service executes the connection environment SQL each time it connects to the database. **Note:** If you enable Sqoop, Sqoop ignores this property.
Transaction SQL	Optional. Enter SQL commands to set the database environment when you connect to the database. The Data Integration Service executes the transaction environment SQL at the beginning of each transaction. **Note:** If you enable Sqoop, Sqoop ignores this property.
SQL Identifier Character	Type of character that the database uses to enclose delimited identifiers in SQL queries. The available characters depend on the database type. Select (None) if the database uses regular identifiers. When the Data Integration Service generates SQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL queries, the service encloses delimited identifiers within this character. **Note:** If you enable Sqoop, Sqoop ignores this property.
Support Mixed-case Identifiers	Enable if the database uses case-sensitive identifiers. When enabled, the Data Integration Service encloses all identifiers within the character selected for the **SQL Identifier Character** property. When the **SQL Identifier Character** property is set to none, the **Support Mixed-case Identifiers** property is disabled. **Note:** If you enable Sqoop, Sqoop honors this property when you generate and execute a DDL script to create or replace a target at run time. In all other scenarios, Sqoop ignores this property.

Property	Description
Use Sqoop Connector	Enables Sqoop connectivity for the data object that uses the JDBC connection. The Data Integration Service runs the mapping in the Hadoop run-time environment through Sqoop. You can configure Sqoop connectivity for relational data objects, customized data objects, and logical data objects that are based on a JDBC-compliant database. Select **Sqoop v1.x** to enable Sqoop connectivity. Default is **None**.

Sqoop Arguments	Enter the arguments that Sqoop must use to connect to the database. Separate multiple arguments with a space.
	To run the mapping on the Blaze engine with the Teradata Connector for Hadoop (TDCH) specialized connectors for Sqoop, you must define the TDCH connection factory class in the Sqoop arguments. The connection factory class varies based on the TDCH Sqoop Connector that you want to use.
	- To use Cloudera Connector Powered by Teradata, configure the following Sqoop argument:
	`–Dsqoop.connection.factories=com.cloudera.connector.teradata.TeradataManagerFactory`
	- To use Hortonworks Connector for Teradata (powered by the Teradata Connector for Hadoop), configure the following Sqoop argument:
	`–Dsqoop.connection.factories=org.apache.sqoop.teradata.TeradataManagerFactory`
	To run the mapping on the Spark engine, you do not need to define the TDCH connection factory class in the Sqoop arguments. The Data Integration Service invokes the Cloudera Connector Powered by Teradata and Hortonworks Connector for Teradata (powered by the Teradata Connector for Hadoop) by default.
	Note: To run the mapping with a generic JDBC connector instead of the specialized Cloudera or Hortonworks connector, you must define the --driver and --connection-manager Sqoop arguments in the JDBC connection. If you define the --driver and --connection-manager arguments in the Read or Write transformation of the mapping, Sqoop ignores the arguments.
	If you do not enter Sqoop arguments, the Data Integration Service constructs the Sqoop command based on the JDBC connection properties.
	On the Hive engine, to run a column profile on a relational data object that uses Sqoop, set the Sqoop argument m to 1. Use the following syntax:
	`-m 1`

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.

- Version of the Kafka messaging broker. You can select one of the following values:

  - 0.8.x.x-0.10.0.x

  - 0.10.1.x-1.0.0

  You can only use one of the versions at a time in a streaming mapping.

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.

Property	Description
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.
Kafka Broker Version	The version of the Kafka messaging broker. You can select one of the following values: - 0.8.x.x-0.10.0.x - 0.10.1.x-1.0.0

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 createConnection
```

On Windows, run the following command:

```
infacmd.bat 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 createConnection -dn <domain name> -un <domain user> -pd <domain password> -cn
<connection name> -cid <connection id> -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*.

MapRStreams Connection

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

When you configure an MapRStreams connection, you configure the path to the MapR Stream and the name of the MapR Stream.

Before you create and use MapRStreams connections in Streaming mappings, complete the required prerequisites. For more information about the prerequisite tasks, see the *Informatica Big Data Management Cluster Integration Guide*.

General Properties

The following table describes the general connection properties for the MapRStreams 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/MapRStreams.

Connection Properties

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

Property	Description
Stream Path and Name	The path that contains the MapR Stream and the name of the MapR Stream. Use the following format: `/stream-path:stream-name`
Connection Timeout (in msec)	Number of milliseconds that the Integration service waits to establish a connection to MapR Stream after which it times out.

Creating a MapRStreams Connection using Informatica Command Line Program

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

On UNIX, run the following command:

```
sh infacmd.sh createConnection
```

On Windows, run the following command:

```
infacmd.bat createConnection
```

Enter connection options in the following format:

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

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

```
infacmd createConnection -dn <domain name> -un <domain user> -pd<password> -cn <connection name> -cid <connection id> -ct MAPRSTREAMS -o connRetryTimeout=10000 streamName=/sample-stream
```

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

Microsoft Azure Data Lake Store Connection

Use the Microsoft Azure Data Lake Store Connection to access Microsoft Azure Data Lake Store tables as targets. You can create and manage an Azure EventHub connection in the Developer tool.

You cannot run a mapping with a Microsoft Data Lake Store as target on a MapR distribution.

Microsoft Azure Data Lake Store Connection Properties

When you set up a Microsoft Azure Data Lake Store connection, you must configure the connection properties.

The following table describes the Microsoft Azure Data Lake Store connection properties:

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	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. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select Microsoft Azure Data Lake Store.

The following table describes the properties for metadata access:

Property	Description
ADLS Account Name	The name of the Microsoft Azure Data Lake Store.
ClientID	The ID of your application to complete the OAuth Authentication in the Active Directory.
Client Secret	The client secret key to complete the OAuth Authentication in the Active Directory.
Directory	The Microsoft Azure Data Lake Store directory that you use to read data or write data. The default is root directory.
AuthEndpoint	The OAuth 2.0 token endpoint from where access code is generated based on based on the Client ID and Client secret is completed.

For more information about creating a client ID, client secret, and auth end point, contact the Azure administrator or visit the Microsoft Azure website at the following link:

https://docs.microsoft.com/en-us/azure/data-lake-store/data-lake-store-authenticate-using-active-directory

APPENDIX B

Data Type Reference

This appendix includes the following topics:

Data Type Reference Overview

Informatica Developer uses the following data types in Streaming mappings:

- Complex file format data types. Streaming mappings can read, process, and write data in complex file formats such as, Avro, JSON, and XML.
- Transformation data types. Set of data types that appear in the transformations. They are internal data types which the Data Integration Service uses to move data across platforms. Transformation data types appear in all transformations in a mapping.

When the Data Integration Service reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When the Data Integration Service writes to a target, it converts the transformation data types to the comparable native data types.

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

Complex File Formats and Transformation Data Types

Streaming mappings can read, process, and write data in complex file formats such as, Avro, JSON, and XML. The data types for these formats appear in the physical data object column properties.

Avro Data Types

The following table lists the Avro data types that Data Integration Service supports and the corresponding transformation data types:

Avro Data Type	Transformation Data Type	Range and Description
array	array	The elements in the array are of string, double, or int data type. The elements in the array are delimited by commas. For example, an array of fruits is represented as [apple,banana,orange]
boolean	integer	The default transformation type for boolean is integer. You can specify string data type with values of True and False. True is equivalent to the integer 1 and False is equivalent to the integer 0.
datetime	date/time	The time stamp format is YYYY-MM-DD HH:MM:SS.SSS. Precision 29, scale 9.
double	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15.
float	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15.
int	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
long	bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0.
map	map	Maps contain key-value pairs and are represented as pairs of strings and integers delimited by the =character. String and integer pairs are delimited by commas. For example, a map of fruits is represented as [1=apple, 2=banana,3=orange].
record	struct	Unlimited number of characters.
string	string	1 to 104,857,600 characters.
struct	struct	Structs are represented as pairs of strings and integers delimited by the : character. String and integer pairs are delimited by commas. For example, a struct of fruits is represented as [1,apple]
union	Corresponding data type in a union of ["primitive_type| complex_type", "null"] or ["null", "primitive_type|complex_type"]	Dependent on primitive or complex data type.

Flat Data Types

The following table lists the Flat data types that Data Integration Service supports and the corresponding transformation data types:

Flat Data Type	Transformation Type	Description
bigint		The elements in the array are of string data type. The elements in the array are delimited by commas. For example, an array of fruits is represented as [apple,banana,orange]
datetime	Date/Time	16 bytes Jan 1, 0001 A.D. to Dec 31, 9999 A.D. Precision of 29, scale of 9 (precision to the nanosecond) Combined date/time value.
number	double	Decimal value with declared precision and scale. Scale must be less than or equal to precision. For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
int	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
nstring	string	1 to 104,857,600 characters
string		1 to 255 characters
timestampWithTZ		Aug. 1, 1947 A.D to Dec. 31, 2040 A.D. -12:00 to +14:00 Precision of 36 and scale of 9. (precision to the nanosecond) Timestamp with Time Zone data type does not support the following time zone regions:- AFRICA_CAIRO- AFRICA_MONROVIA- EGYPT- AMERICA_MONTREAL

JSON Data Types

The following table lists the JSON data types and the corresponding transformation data types:

JSON Data Type	Transformation Data Type	Range and Description
array	array	The elements in the array are of string, double, or int data type. The elements in the array are delimited by commas. For example, an array of fruits is represented as [apple,banana,orange]
boolean	integer	The default transformation type for boolean is integer. You can specify string data type with values of True and False. True is equivalent to the integer 1 and False is equivalent to the integer 0.
datetime	date/time	The time stamp format is YYYY-MM-DD HH:MM:SS.SSS. Precision 29, scale 9.
double	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15.
int	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
long	bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
string	string	1 to 255 characters
struct	struct	Structs are represented as pairs of strings and integers delimited by the : character. String and integer pairs are delimited by commas.

XML Data Types

The following table lists the XML data types that Data Integration Service supports and the corresponding transformation data types:

Native XML DataType	Transformation Data Type	Range and Description
anyURI	string	1 to 255 characters
array	array	The elements in the array are of string, double, or int data type. The elements in the array are delimited by commas. For example, an array of fruits is represented as [apple,banana,orange]
boolean	integer	The default transformation type for boolean is integer. You can specify string data type with values of True and False. True is equivalent to the integer 1 and False is equivalent to the integer 0.

Native XML DataType	Transformation Data Type	Range and Description
byte	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
date	date/time	The time stamp format is YYYY-MM-DD HH:MM:SS.SSS. Precision 29, scale 9.
dateTime	date/time	The time stamp format is YYYY-MM-DD HH:MM:SS.SSS. Precision 29, scale 9.
decimal	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15
double	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15
everythingElse	string	1 to 255 characters
float	double	-1.79769313486231570E+308 to +1.79769313486231570E+308. Precision 15
integer	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
int	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
long	bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
nonPositiveInteger	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
negativeInteger	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
nonNegativeInteger	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
positiveInteger	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
QName	string	1 to 255 characters
short	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
string	string	1 to 255 characters
struct	struct	Structs are represented as pairs of strings and integers delimited by the : character. String and integer pairs are delimited by commas. For example, a struct of fruits is represented as [1,apple]
time	string	1 to 255 characters

Native XML DataType	Transformation Data Type	Range and Description
unsignedInt	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
unsignedShort	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
unsignedByte	integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
unsignedLong	bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0

If you have multiple date and time formats in the data payload that you read from, they must be must be converted to Date/Time data type. Get the date and time stamp data in string format and then use an expression transformation to convert the string to Date/Time.

For example, the following expression converts the string in the `shipment_date` port to a date value:

```
TO_DATE(shipment_date,'YYYY-MM-DD')
```

The Data Integration Service uses the following default date/time format of applications:

```
MM/DD/YYYY HH24:MI:SS
```

You can set this property on the **Advanced** tab of the application, when the application contains a mapping.

You can set Default date time format property for the Data Integration Service in the **Preferences** > **Informatica** > **Run Configurations** > **Mapping** > **Advanced** tab.

Note: When you perform a lookup on timestamp data or write timestamp data to a target in a Streaming mapping, the Spark engine reads or writes the time only up to the microseconds. This occurs because the default size of the timestamp field is 8 bytes.

Sqoop Data Types

When you use Sqoop, some variations apply in the processing. Sqoop supports a subset of data types that database vendors support.

Microsoft SQL Server Data Types

Informatica supports the following Microsoft SQL Server data types when you use Sqoop:

- Bigint
- Bit
- Char
- Datetime
- Datetime2
- Decimal
- Float

- Image
- INT
- Money
- Nchar
- Ntext
- Numeric
- Nvarchar
- Real
- Smalldatetime
- Smallint
- Smallmoney
- Sysname
- Timestamp
- Tinyint
- Uniqueidentifier
- Varbinary
- Varchar

Rules and Guidelines for Sqoop Microsoft SQL Server Data Types

Consider the following rules and guidelines when you configure Microsoft SQL Server data types when you use Sqoop in a streaming mapping:

- Use the Varbinary data type if the target table that you write to has a column of binary data type because Spark DataFrame does not have a corresponding data type to match its native type and the mapping might throw a runtime exception.
- Use the Varbinary data type only if you have imported the table while creating the relational target or you selected the **Create or replace table at runtime** property in the Advance properties. Spark DataFrame does not have a corresponding data type to match its native type.
- If you export timestamp data, an error occurs as the timestamp data is automatically generated by the database. Use the datetime data type.

Oracle Data Types

Informatica supports the following Oracle data types when you use Sqoop:

- Blob
- Char(L)
- Date
- Float
- Long
- Long Raw
- Nchar
- Nclob
- Number

- Number(P,S)
- Nvarchar2
- Raw
- Timestamp
- Varchar
- Varchar2

APPENDIX C

Sample Files

This appendix includes the following topic:

Sample Files

The data objects in a streaming mapping read and write data in XML, JSON, Avro, CSV format. The following examples contain samples for each schema format.

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:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="shiporder">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="orderperson" type="xs:string"/>
      <xs:element name="shipto">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="address" type="xs:string"/>
            <xs:element name="city" type="xs:string"/>
            <xs:element name="country" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="item" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="title" type="xs:string"/>
            <xs:element name="note" type="xs:string" minOccurs="0"/>
            <xs:element name="quantity" type="xs:positiveInteger"/>
            <xs:element name="price" type="xs:decimal"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="orderid" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Sample JSON Schema

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:

```
{
                "GlossaryCont": {
                                "title": {
                                                "string": "glossaryContTitle0"
                                },
                                "glossDiv": {
                                                "title": "glossaryDivTitle0",
                                                "glossList": {
                                                                "glossEntry": {
                                                                                "id":
"ID0",

"sortAs": "sortAS0",

"glossTerm": "GlossTerm0",

"acronym": "Acronym0",

"abbrev": "Abbrev0",

"glossSee": "GlossSee0",

"glossInteger": 0,

"glossDouble": 0.234,

"glossLong": 100000000000000000,

"glossDef": {

"para": "para0",

"glossSeeAlso": ["glossSeeAlso_0_0"]
                                                                                }
                                                                }
                                                }
                                }
                }
}
```

Sample Avro Schema

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 schema in a .avsc file.

The following file is a sample Avro schema:

```
{
  "type" : "record",
  "name" : "Tree",
  "fields" : [
{"name" : "children", "type" : ("type" : "array", "items": "Tree"}}
]}
```

Sample Flat Format Schema

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

The following file is a sample CSV file:

```
acc_no,acc_type
1,Savings
2,Savings
```

INDEX

A

B

C

D

H

I

J

K

M

P

R

S

T

W