Informatica®

Informatica® Cloud Data Integration

Kafka Connector

Informatica Cloud Data Integration Kafka Connector
February 2025

Publication Date: 2025-02-10

Table of Contents

Preface

Use *Kafka Connector* to learn how to read from or write to Kafka by using Cloud Data Integration. Learn to create a connection, develop and run mappings, and mapping tasks in Cloud Data Integration.

Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit https://docs.informatica.com.

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

Informatica Intelligent Cloud Services web site

You can access the Informatica Intelligent Cloud Services web site at http://www.informatica.com/cloud. This site contains information about Informatica Cloud integration services.

Informatica Intelligent Cloud Services Communities

Use the Informatica Intelligent Cloud Services Community to discuss and resolve technical issues. You can also find technical tips, documentation updates, and answers to frequently asked questions.

Access the Informatica Intelligent Cloud Services Community at:

https://network.informatica.com/community/informatica-network/products/cloud-integration

Developers can learn more and share tips at the Cloud Developer community:

https://network.informatica.com/community/informatica-network/products/cloud-integration/cloud-developers

Informatica Intelligent Cloud Services Marketplace

Visit the Informatica Marketplace to try and buy Data Integration Connectors, templates, and mapplets:

https://marketplace.informatica.com/

Data Integration connector documentation

You can access documentation for Data Integration Connectors at the Documentation Portal. To explore the Documentation Portal, visit https://docs.informatica.com.

Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit https://search.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 Intelligent Cloud Services Trust Center

The Informatica Intelligent Cloud Services Trust Center provides information about Informatica security policies and real-time system availability.

You can access the trust center at https://www.informatica.com/trust-center.html.

Subscribe to the Informatica Intelligent Cloud Services Trust Center to receive upgrade, maintenance, and incident notifications. The Informatica Intelligent Cloud Services Status page displays the production status of all the Informatica cloud products. All maintenance updates are posted to this page, and during an outage, it will have the most current information. To ensure you are notified of updates and outages, you can subscribe to receive updates for a single component or all Informatica Intelligent Cloud Services components. Subscribing to all components is the best way to be certain you never miss an update.

To subscribe, on the Informatica Intelligent Cloud Services Status page, click **SUBSCRIBE TO UPDATES**. You can choose to receive notifications sent as emails, SMS text messages, webhooks, RSS feeds, or any combination of the four.

Informatica Global Customer Support

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

To find online support resources on the Informatica Network, click **Contact Support** in the Informatica Intelligent Cloud Services Help menu to go to the **Cloud Support** page. The **Cloud Support** page includes system status information and community discussions. Log in to Informatica Network and click **Need Help** to find additional resources and to contact Informatica Global Customer Support through email.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at https://www.informatica.com/services-and-training/support-services/contact-us.html.

CHAPTER 1

Introduction to Kafka Connector

You can use Kafka connector to connect to Kafka from Data Integration.

You can use Kafka topics as sources and targets in mappings or mapping tasks. When you use Kafka topics in mappings, you can configure properties specific to Kafka. Kafka Connector uses the Kafka Producer and Consumer APIs to connect to Kafka. Use Kafka Connector to import a Kafka topic in Avro or JSON format in mappings to read and write primitive data types.

You can switch the mapping to advanced mode to include transformations and functions that enable advanced functionality. You can read and write primitive and hierarchical data types for Avro and JSON format in a mapping in advanced mode.

The advanced cluster can be hosted on Amazon Web Services, Google Cloud Platform, or Microsoft Azure environment.

Kafka Connector assets

Create assets in Data Integration to integrate data using Kafka connector.

You can perform insert operations on a Kafka target.

When you use Kafka connector, you can include the following Data Integration assets:

- Mapping
- Mapping task

For more information about configuring assets, see *Mappings* and *Tasks* in the Data Integration documentation.

Administration of Kafka Connector

Kafka Connector uses Confluent version 5.5.6 and Kafka client version 3.4.0.

Before you use Kafka Connector, complete the following prerequisite tasks:

- You must not enable multiple Hadoop distribution packages in your organization.
- To run a mapping in advanced mode, verify that the Secure Agent is associated with an advanced cluster configuration.

- To read data from or write data to a Kafka topic in Avro format without schema registry, you must configure the JVM Options and INFA_DEBUG property for the Secure Agent to enable Avro data format.

Configuring the Secure Agent to import Avro schema without schema registry

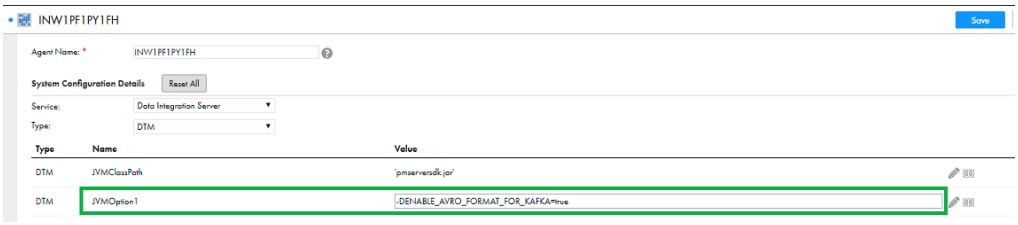
If you configure a mapping to read data from or write data to a Kafka topic in Avro format without schema registry, you must configure the JVM Options and INFA_DEBUG property for the Secure Agent to enable Avro data format.

To configure the Secure Agent and successfully import a Kafka topic in Avro format, perform the following steps:

1. In Administrator, select **Runtime Environments**.
2. Select the Secure Agent for which you want to configure from the list of available Secure Agents.
3. In the upper-right corner, click **Edit**.
4. In the **System Configuration Details** section, select the **Type** as **DTM** for the Data Integration Service.
5. Edit the **JVMOption1** and add the following value:

   ```
   -DENABLE_AVRO_FORMAT_FOR_KAFKA=true
   ```

   The following image shows the System Configuration Details section and the configured JVMOption1:



6. Select the **Type** as **Platform** for the Data Integration Service.
7. Edit the INFA_DEBUG property and add the following value:

   ```
   -DENABLE_AVRO_FORMAT_FOR_KAFKA=true
   ```

   The following image shows the System Configuration Details section and the configured INFA_DEBUG property:

8. Click **Save**.
9. Restart the Secure Agent.

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 can use Kafka Connector to read network activity data from Kafka topics in JSON format and write the data to a target for processing the network activity data.

CHAPTER 2

Kafka connections

Create a Kafka connection to access an Apache Kafka broker and read data from and write data to Apache Kafka brokers. You can use Kafka connections in mappings and mapping.

Kafka connection properties

When you set up a Kafka connection, configure the connection properties.

The following table describes the Kafka connection properties:

Property	Description
Connection Name	Name of the connection. The name is not case sensitive. It 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: ~ ` ! $ % ^ & * ( ) - + = { [}] | \ : ; " ' < , > . ? /
Description	Optional. Description that you use to identity the connection. The description cannot exceed 4,000 characters.
Type	The Kafka connection type. If you do not see the connection type, go to the **Add-On Connectors** page in Administrator to install the connector.
Runtime Environment	Name of the runtime environment where you want to run tasks. Specify a Secure Agent or a serverless runtime environment for a mapping that runs on the advanced cluster.
Kafka Broker List	Comma-separated list of the Kafka brokers. To list a Kafka broker, use the following format: *<HostName>:<PortNumber>* **Note:** When you connect to a Kafka broker over SSL, you must specify the fully qualified domain name for the host name. Otherwise, the test connection fails with SSL handshake error.
Retry Timeout	Optional. Number of seconds after which the Secure Agent attempts to reconnect to the Kafka broker to read or write data. Default is 180 seconds.

Property	Description
Kafka Broker Version	Kafka message broker version. The only valid value is Apache 0.10.1.1 and above.
Additional Connection Properties	Optional. Comma-separated list of additional configuration properties of the Kafka producer or consumer.
Confluent Schema Registry URL	Location and port of the Confluent schema registry service to access Avro sources and targets in Kafka. To list a schema registry URL, use the following format: <https>://<*HostName or IP*>:<*PortNumber*> or <http>://<*HostName or IP*>:<*PortNumber*> Example for the schema registry URL: https://kafkarnd.informatica.com:8082 or http://10.65.146.181:8084 Applies only when you import a Kafka topic in Avro format that uses the Confluent schema registry to store the metadata.
SSL Mode	Required. Determines the encryption type to use for the connection. You can choose a mode from the following SSL modes: - Disabled. Establishes an unencrypted connection to the Kafka broker. - One-way. Establishes an encrypted connection to the Kafka broker using truststore file and truststore password. - Two-way. Establishes an encrypted connection to the Kafka broker using truststore file, truststore password, keystore file, and keystore password.
SSL TrustStore File Path	Required when you use the one-way or two-way SSL mode. Absolute path and file name of the SSL truststore file that contains the SSL certificate to connect to the Kafka broker.
SSL TrustStore Password	Required when you use the one-way or two-way SSL mode. Password for the SSL truststore.
SSL KeyStore File Path	Required when you use the two-way SSL mode. Absolute path and file name of the SSL keystore file that contains private keys and certificates to connect to the Kafka broker.
SSL KeyStore Password	Required when you use the two-way SSL mode. Password for the SSL keystore.
Additional Security Properties	Optional. Comma-separated list of additional configuration properties to connect to the Kafka broker in a secure way. If you specify two different values for the same property in **Additional Connection Properties** and **Additional Security Properties**, the value in **Additional Security Properties** overrides the value in **Additional Connection Properties**.

Schema Registry Security Configuration Properties

When you configure the **Schema Registry URL** connection property, you can configure the schema registry security configuration properties. These properties apply only to mappings in advanced mode. You can

configure one-way SSL, two-way SSL, and basic authentication to connect to the Confluent schema registry in a secure way.

The following table describes the security properties for the Kafka connection when you use the Confluent schema registry:

Property	Description
SSL Mode Schema Registry[1]	Required. Determines the encryption type to use for the connection. You can choose a mode from the following SSL modes: - Disabled. Establishes an unencrypted connection to the Confluent schema registry. - One-way. Establishes an encrypted connection to the Confluent schema registry using truststore file and truststore password. - Two-way. Establishes an encrypted connection to the Confluent schema registry using truststore file, truststore password, keystore file, and keystore password.
SSL TrustStore File Path Schema Registry[1]	Required when you use the one-way or two-way SSL mode. Absolute path and file name of the SSL truststore file that contains the SSL certificate to connect to the Confluent schema registry.
SSL TrustStore Password Schema Registry[1]	Required when you use the one-way or two-way SSL mode. Password for the SSL truststore.
SSL KeyStore File Path Schema Registry[1]	Required when you use the two-way SSL mode. Absolute path and file name of the SSL keystore file that contains private keys and certificates to connect to the Confluent schema registry.
SSL KeyStore Password Schema Registry[1]	Required when you use the two-way SSL mode. Password for the SSL keystore.
Additional Security Properties Schema Registry[2]	Optional. Comma-separated list of additional security properties to connect to the Confluent schema registry in a secure way. For example, when you configure basic authentication to establish a secure communication with Confluent schema registry, specify the following value: `basic.auth.credentials.source=USER_INFO,basic.auth.user.info=<username>:<password>` If you specify two different values for the same property in **Additional Connection Properties** and **Additional Security Properties Schema Registry**, the value in **Additional Security Properties Schema Registry** overrides the value in **Additional Connection Properties**.

[1] Applies only to mappings in advanced mode.

[2] Applies to both mappings and mappings in advanced mode.

Configuring Confluent schema registry for Avro format

When you read Avro data from or write Avro data to a Kafka topic in a mapping, you can configure the Kafka connection to use the Confluent schema registry to import Avro metadata.

Confluent schema registry provides a serving layer for your metadata to store and retrieve Avro schemas. Schema registry stores all the versions of Avro schemas based on a subject name strategy and provides multiple compatibility settings. Schema registry allows evolution of schemas according to the configured compatibility settings and expanded support for these schema types.

Schema registry provides Avro serializers and deserializers that Kafka Connector uses to handle schema storage and retrieval of Kafka messages in Avro format.

Schema registry is available separately from your Kafka brokers. The Kafka producers and consumers connect to Kafka to publish and read data from topics. Concurrently, the Kafka Connector connects to the schema registry to send and retrieve schemas that describe the data models for the Kafka messages.

Configuring basic authentication for Confluent schema registry in a mapping

You can configure basic authentication to establish a secure communication with Confluent schema registry using a valid user name and password.

Configure the Kafka connection

Perform the following tasks to configure the Kafka connection to enable basic authentication with the Confluent schema registry:

1. In **Administrator**, select **Connections**.
2. Select a Kafka connection for which you want to enable basic authentication with the Confluent schema registry.
3. Click **Edit**.
4. In the **Schema Registry Security Configuration Properties** section of the Kafka connection properties, specify the following value in the **Additional Security Properties Schema Registry** property:
   `basic.auth.credentials.source=USER_INFO,basic.auth.user.info=<username>:<password>`
5. Click **Save** to save the connection.

Configure the Secure Agent

Configure the JVM Options and INFA_DEBUG property for the Secure Agent to enable basic authentication with the Confluent schema registry and to successfully import the Avro metadata from the Confluent schema registry.

To configure the Secure Agent and successfully import the Avro metadata, perform the following steps:

1. In Administrator, select **Runtime Environments**.
2. Select the Secure Agent for which you want to configure from the list of available Secure Agents.
3. In the upper-right corner, click **Edit**.
4. In the **System Configuration Details** section, select the **Type** as **DTM** for the Data Integration Service.

5. Edit the **JVMOption2** and add the following value:
   `-Dbasic.auth.user.info=<username>:<password>`
6. Select the **Type** as **Platform** for the Data Integration Service.
7. Edit the INFA_DEBUG property and add the following value:
   `-Dbasic.auth.user.info=<username>:<password>`
8. Click **Save**.
9. Restart the Secure Agent.

Configuring one-way SSL authentication for Confluent schema registry in a mapping

You can configure SSL authentication to establish one-way secure communication with Confluent schema registry.

Import the Confluent schema registry truststore certificate

Import the Confluent schema registry truststore certificate and ensure that the certificates are in the `.jks` format.

1. Download the latest `.jks` truststore file from the secured Kafka broker associated with a secured schema registry.
2. Extract the certificate from the `.jks` truststore file in PEM format.
   The certificate is exported in `.cer` format.
3. When prompted, specify the password for the truststore file.
4. Import the `.cer` certificate into the cacerts file located in the jdk directory available at one or more of the following locations within your Secure Agent installation:
   - `<Secure Agent installation directory>/jdk/jre/lib/security`
   - `<Secure Agent installation directory>/jdk/lib/security`
   - `<Secure Agent installation directory>/jdk8/jre/lib/security`

   Ensure to add the certificate to all the available directories.
5. When prompted, specify the password for the `cacerts` file.
6. If there is a jdk directory within the `<Secure agent installation directory>\apps` folder, navigate to the following directory and import the `.cer` certificate into the `cacerts` file located in the jdk directory available at one or more of the following locations within your Secure Agent installation:
   - `<Secure agent installation directory>\apps\jdk\zulu8<latest_version>\jre\lib\security`
   - `<Secure agent installation directory>\apps\jdk\zulu17<latest_version>\lib\security`
7. After you import the `cacerts` file, verify the entry of the `.cer` certificate.
8. Restart the Secure Agent.

Configure the Kafka connection

Configure the Kafka connection to enable one-way SSL authentication with the Confluent schema registry:

1. In **Administrator**, select **Connections**.

2. Select a Kafka connection for which you want to configure one-way SSL authentication with the Confluent schema registry.
3. Click **Edit**.
4. In the Kafka connection properties, select the **SSL Mode** as **One-way**.
5. Specify the **SSL TrustStore File Path** and the **SSL TrustStore Password**.
6. Click **Save** to save the connection.

Configure the Secure Agent

Configure the JVM Options and INFA_DEBUG property for the Secure Agent to configure one-way SSL authentication with the Confluent schema registry and to successfully import the Avro metadata from the Confluent schema registry.

To configure the Secure Agent and successfully import the Avro metadata, perform the following steps:

1. In Administrator, select **Runtime Environments**.
2. Select the Secure Agent for which you want to configure from the list of available Secure Agents.
3. In the upper-right corner, click **Edit**.
4. In the **System Configuration Details** section, select the **Type** as **DTM** for the Data Integration Service.
5. Edit the following JVM options and add the following values:

Property	Value
JVMOption3	-Dsr.sslTruststoreFilePath=<schema registry truststore certificate file path>/ schema_registry.truststore.jks
JVMOption4	-Dsr.sslTruststorePassword=<password for the schema registry truststore certificate>

6. Select the **Type** as **Platform** for the Data Integration Service.
7. Edit the INFA_DEBUG property and add the following space separated values:
   `-Dsr.sslTruststoreFilePath=<schema registry truststore certificate file path>/ schema_registry.truststore.jks -Dsr.sslTruststorePassword=<password for the schema registry truststore certificate>`
8. Click **Save**.
9. Restart the Secure Agent.

Connecting to a kerberised Kafka cluster on Linux

To read from or write to a Kerberised Kafka cluster that runs on Linux operating system, configure the default realm, KDC, and Kafka advanced source or target properties.

You can configure Kerberos authentication for a Kafka client by placing the required Kerberos configuration files on the Secure Agent machine and specifying the required JAAS configuration in the Kafka connection. The JAAS configuration defines the keytab and principal details that the Kafka broker must use to authenticate the Kafka client.

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

1. Ensure that you have the `krb5.conf` file for the Kerberised Kafka cluster.

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 Kafka keytab  file>/<Kafka 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.
   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. Configure the following properties:

Kafka connection

Configure the **Additional Connection Properties** property in a Kafka connection and specify the value in the following format:

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

Sources

Configure the **Consumer Configuration Properties** property in the advanced source properties to override the value specified in the **Additional Connection Properties** property in a Kafka connection. Specify the value in the following format:

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

Targets

Configure the **Producer Configuration Properties** property in the advanced target properties to override the value specified in the **Additional Connection Properties** property in a Kafka connection. Specify the value in the following format:

```
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, configure the following properties:

Kafka connection

Configure the **Additional Connection Properties** property in a Kafka connection and specify the value in the following format:

```
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>";
```

Sources

Configure the **Consumer Configuration Properties** property in the advanced source properties to override the value specified in the **Kerberos Configuration Properties** property in a Kafka connection. Specify the value in the following format:

```
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>";
```

Targets

Configure the **Producer Configuration Properties** property in the advanced target properties to override the value specified in the **Kerberos Configuration Properties** property in a Kafka connection. Specify the value in the following format:

```
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>";
```

Connecting to a kerberised Kafka cluster on Windows

To read from or write to a Kerberised Kafka cluster that runs on Windows operating system, configure the default realm, KDC, and Kafka advanced source or target properties.

You can configure Kerberos authentication for a Kafka client by placing the required Kerberos configuration files on the Secure Agent machine and specifying the required JAAS configuration in the Kafka connection. The JAAS configuration defines the keytab and principal details that the Kafka broker must use to authenticate the Kafka client.

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

1. Ensure that you have the `krb5.ini` file for the Kerberised Kafka cluster.
2. Configure the default realm and KDC. If the default `C:\Windows\krb5.ini` file is not configured or you want to change the configuration, add the following lines to the `C:\Windows\krb5.ini` 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. In the **System Configuration Details** section of the Secure Agent, select the **Type** as **Tomcat JRE** for the Data Integration Server. Edit **JRE_OPTS** as **'-Xrs -Djava.security.krb5.conf=C:\Windows\krb5.ini'**
4. 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="<Kafka keytab file directory>\\<Kafka 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.
   Informatica recommends that you place the files in a location that is accessible to all the nodes in the cluster. Example: `C:\\Windows` or `\temp`
   c. Configure the following properties:

Kafka connection

Configure the **Additional Connection Properties** property in a Kafka connection and specify the value in the following format:

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

Sources

Configure the **Consumer Configuration Properties** property in the advanced source properties to override the value specified in the **Additional Connection Properties** property in a Kafka connection. Specify the value in the following format:

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

Targets

Configure the **Producer Configuration Properties** property in the advanced target properties to override the value specified in the **Additional Connection Properties** property in a Kafka connection. Specify the value in the following format:

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

5. To embed the JAAS configuration in the `sasl.jaas.config` configuration property, configure the following properties:

Kafka connection

Configure the **Additional Connection Properties** property in a Kafka connection and specify the value in the following format:

```
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="<Kafka keytab file directory>\\<Kafka keytab file name>"
client=true principal="<principal_name>";
```

Sources

Configure the **Consumer Configuration Properties** property in the advanced source properties to override the value specified in the **Kerberos Configuration Properties** property in a Kafka connection. Specify the value in the following format:

```
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="<Kafka keytab file directory>\\<Kafka keytab file name>"
client=true principal="<principal_name>";
```

Targets

Configure the **Producer Configuration Properties** property in the advanced target properties to override the value specified in the **Kerberos Configuration Properties** property in a Kafka connection. Specify the value in the following format:

```
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="<Kafka keytab file directory>\\<Kafka keytab file name>"
client=true principal="<principal_name>";
```

Configuring SASL PLAIN authentication for a Kafka broker

In the Kafka connection, you can configure PLAIN security for the Kafka broker to connect to a Kafka broker. To read data from or write data to a Kafka broker with SASL PLAIN authentication, configure the Kafka

connection properties. To override the properties defined in the Kafka connection, you can configure the advanced source or target properties.

You can configure SASL PLAIN authentication so that the Kafka broker can authenticate the Kafka producer and the Kafka consumer. Kafka uses the Java Authentication and Authorization Service (JAAS) for SASL PLAIN authentication. To enable SASL PLAIN authentication, you must specify the SASL mechanism as PLAIN. You must also provide the formatted JAAS configuration that the Kafka broker must use for authentication. The JAAS configuration defines the username, password, that the Kafka broker must use to authenticate the Kafka client.

Configure the following properties:

Kafka connection

Configure the **Additional Connection Properties** or **Additional Security Properties** property in the Kafka connection and specify the value in the following format:

```
security.protocol=SASL_SSL,sasl.mechanism=PLAIN,sasl.jaas.config=org.apache.kafka.com
mon.security.plain.PlainLoginModule required username=<username> password=<password>
```

In the **Security Configuration Section**, select **One-Way** as the **SSL Mode** and specify the SSL TrustStore File Path and SSL TrustStore Password.

Sources

Configure the **Consumer Configuration Properties** property in the advanced source properties to override the value that you specified in the **Additional Connection Properties** property in the Kafka connection. Specify the value in the following format:

```
security.protocol=SASL_SSL,sasl.mechanism=PLAIN,sasl.jaas.config=org.apache.kafka.com
mon.security.plain.PlainLoginModule required username=<username> password=<password>
```

Targets

Configure the **Producer Configuration Properties** property in the advanced target properties to override the value that you specified in the **Additional Connection Properties** property in the Kafka connection. Specify the value in the following format:

```
security.protocol=SASL_SSL,sasl.mechanism=PLAIN,sasl.jaas.config=org.apache.kafka.com
mon.security.plain.PlainLoginModule required username=<username> password=<password>
```

Configuring SASL PLAIN authentication for an Azure Event Hub Kafka broker

In the Kafka connection, you can configure PLAIN security for the Kafka broker to connect to an Azure Event Hub Kafka broker. When you connect to an Azure Event Hub Kakfa broker, the password defines the endpoint URL that contains the fully qualified domain name (FQDN) of the Event Hub namespace, shared access key name, and shared access key required to connect to an Azure Event Hub Kafka broker.

To connect to an Azure Event Hub Kafka broker, configure any of the above properties and specify the value in the following format:

```
security.protocol=SASL_SSL,sasl.mechanism=PLAIN,sasl.jaas.config=org.apache.kafka.common.
security.plain.PlainLoginModule required username="$ConnectionString"
password="Endpoint=sb://<FQDN>/;SharedAccessKeyName=<key name>;SharedAccessKey=<shared
access key>=";
```

Connecting to Amazon Managed Streaming for Apache Kafka

In the Kafka connection, you can configure PLAINTEXT or TLS encryption to connect to an Amazon Managed Streaming for Apache Kafka broker. To read data from or write data to an Amazon Managed Streaming for Apache Kafka broker, configure the Kafka connection properties.

Configure the **Kafka Broker List** property in the Kafka connection and specify the comma-separated list of Kafka brokers that you want to connect to in the following format:

```
<HostName>:<PortNumber>
```

Configure TLS encryption to securely connect the Kafka broker to the Kafka producer and the Kafka consumer. To configure TLS encryption for an Amazon Managed Streaming for Apache Kafka broker, configure the following properties:

Property	Values
Additional Connection Properties	`security.protocol=SSL`
SSL Mode	One-way or Two-way.
SSL TrustStore File Path	Required when you use the one-way or two-way SSL mode. Absolute path and file name of the SSL truststore file.
SSL TrustStore Password	Required when you use the one-way or two-way SSL mode. Password for the SSL truststore.
SSL KeyStore File Path	Required when you use the two-way SSL mode. Absolute path and file name of the SSL keystore file that contains private keys and certificates that the Kafka broker validates against the Kafka cluster certificate.
SSL KeyStore Password	Required when you use the two-way SSL mode. Password for the SSL keystore.

When you run a mapping that runs on an advanced cluster and connect to an Amazon Managed Streaming for Apache Kafka broker, configure the Kafka broker using SASL_SSL authentication with Salted Challenge Response Authentication Mechanism (SCRAM). To read data from or write data to an Amazon Managed Streaming for Apache Kafka broker with SASL_SSL authentication, configure the following properties:

Property	Values
Additional Connection Properties	`security.protocol=SASL_SSL,sasl.mechanism=SCRAM-SHA-512,sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule required username="<username>" password="<password>";`
SSL Mode	One-way or Two-way.
SSL TrustStore File Path	Required when you use the one-way or two-way SSL mode. Absolute path and file name of the SSL truststore file.

Property	Values
SSL TrustStore Password	Required when you use the one-way or two-way SSL mode. Password for the SSL truststore.
SSL KeyStore File Path	Required when you use the two-way SSL mode. Absolute path and file name of the SSL keystore file that contains private keys and certificates that the Kafka broker validates against the Kafka cluster certificate.
SSL KeyStore Password	Required when you use the two-way SSL mode. Password for the SSL keystore.

CHAPTER 3

Mappings for Kafka

When you configure a mapping, you describe the flow of data from the source to the target.

A mapping defines reusable data flow logic that you can use in mapping tasks.

When you create a mapping, you define the Source and Target to represent a Kafka object. Use the Mapping Designer in Data Integration to add the Source and Target in the mapping canvas and configure the Kafka source and target properties.

In advanced mode, the Mapping Designer updates the mapping canvas to include transformations and functions that enable advanced functionality.

You can use Monitor to monitor the jobs.

Kafka sources in mappings

To read messages from a Kafka topic, configure a Kafka object as the Source transformation in a mapping.

Specify the name and description of the Kafka source. Configure the source and advanced properties for the source object.

The following table describes the source properties that you can configure for a Kafka source:

Property	Description
Connection	Name of the active Kafka source connection.
Source Type	Type of the Kafka source objects available. Select **Single Object**. You cannot read data from multiple objects. To define a parameter, select **Parameter** as the source type, and then specify the parameter in the Parameter property. **Note:** When you define a parameter, you must import the Kafka source in Binary or JSON format.
Object	Name of the Kafka source object based on the source type selected.
Parameter	Select a parameter for the source object, or click **New Parameter** to define a new parameter for the source object. The **Parameter** property appears only if you select Parameter as the source type.

Property	Description
Format	The file format to read data from a Kafka topic. You can select from the following file format types: - None. Reads data from the Kafka topics in binary format. - Avro. Reads data from the Kafka topics in Avro format that use the binary encoding type. - JSON. Reads data from the Kafka topics in JSON format. - Discover Structure[2]. Determines the underlying patterns in a sample file and auto-generates a model for files with the same data and structure. To configure the formatting options for the file, click **Formatting Options**. For more information about format options, see "Formatting options for Kafka topics" on page 28. **Note:** You can read Snappy compressed records from the broker when you run the mapping in advanced mode.
Preview Data	Preview the data of the Kafka source object. When you preview data, Data Integration displays the first 10 rows. By default, Data Integration displays the fields in native order. To display the fields in alphabetical order, enable the **Display source fields in alphabetical order** option.
Intelligent Structure Model[2]	Applicable to the Discover Structure format type. Select the intelligent structure model to determine the underlying patterns and structures of the input that you provide for the model and create a model that can be used to transform, parse, and generate output groups. Select one of the following options to associate a model with the transformation: - Select. Select an existing model. - New. Create a new model. Select **Design New** to create the model. Select **Auto-generate from sample file** for Intelligent Structure Discovery to generate a model based on sample input that you select. For more information, see *Components*.
[1] Doesn't apply to mappings in advanced mode. [2] Applies only to mappings in advanced mode.	

The following table describes the advanced properties that you can configure for a Kafka source:

Property	Description
Start position offset	The position of the Kafka consumer from where the Kafka Connector 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 all the messages available on the Kafka topic from the beginning. - Latest[1]. Read messages received by the Kafka topic after the mapping has been deployed.
Connector Mode	Specifies the mode to read data from the Kafka source. You can select one of the following modes: - Batch. The Secure Agent reads the messages available in a Kafka topic based on the position offset you specify. After the Secure Agent reads the data, the mapping terminates. When you configure a mapping for recovery, the Secure Agent reads data from the last checkpoint and stops reading data till the offset when the mapping recovery started. - Realtime[1]. The Secure Agent reads the messages available in a Kafka topic in real-time. The Secure Agent continues to read messages from the Kafka topic and does not terminate the mapping. You must manually terminate the mapping task.

Property	Description
Custom Start Position Timestamp	The time in GMT from when the Secure Agent starts reading Kafka messages from a Kafka topic. Specify the time in the following format: `yyyy-mm-dd hh:mm:ss[.fff]` The milliseconds are optional. This property applies only when you set the Start position offset property to **Custom**. You can parameterize this property using an in-out or input parameter in a mapping in advanced mode. Specify the parameter in the following format: - In-out parameter: `$$<parameter_name>` - Input parameter: `$<parameter_name>$`
Topic Pattern	A regular expression pattern for the topic name that you want to read from. Use the regular expression syntax guidelines to specify the pattern.
Polling Interval	The time interval, in milliseconds, that the agent waits before retrieving records from the Kafka broker. You can enter a positive integer value. The default is 1000 milliseconds. The polling interval applies only to the mapping in which you set the value. To set the polling interval for all mappings within the organization, you need to set the `pollIntervalInMilliseconds` property as a JVM option under the DTM type in the Secure Agent properties.
Consumer Configuration Properties	The configuration properties for the Kafka consumer. These properties override the parameters you specified in the **Additional Connection Properties** or **Additional Security Properties** fields in the Kafka connection. For example, you can configure the `group.id` consumer property at source to enable incremental load when you read from Kafka topics in a mapping in advanced mode. For more information about incremental load and how to configure the `group.id` consumer property, see "Configure incremental load to read from Kafka topics" on page 31. For more information about Kafka consumer configuration properties, see the Kafka documentation.
[1] Doesn't apply to mappings in advanced mode.	

You can set the tracing level in the advanced properties to determine the amount of details that logs contain for a mapping. Tracing level is not applicable to mappings in advanced mode.

The following table describes the tracing levels that you can configure:

Property	Description
Terse	The Secure Agent logs initialization information, error messages, and notification of rejected data.
Normal	The Secure Agent logs initialization and status information, errors encountered, and skipped rows due to transformation row errors. Summarizes mapping results, but not at the level of individual rows.

Property	Description
Verbose Initialization	In addition to normal tracing, the Secure Agent logs additional initialization details, names of index and data files used, and detailed transformation statistics.
Verbose Data	In addition to verbose initialization tracing, the Secure Agent logs each row that passes into the mapping. Also notes where the Secure Agent truncates string data to fit the precision of a column and provides detailed transformation statistics. When you configure the tracing level to verbose data, the Secure Agent writes row data for all rows in a block when it processes a transformation.

Configure a mapping to read data from Kafka in real-time

You can configure a mapping to read data from a Kafka topic in real-time.

To read data from a Kafka topic in real-time, perform the following tasks:

1. In the Kafka source advanced properties, configure the **Connector Mode** as **Realtime**.
2. Enable message recovery in the **Advanced Session Properties** section on the **Runtime Options** tab of the mapping task.
   For more information on how to configure the message recovery, see "Configure message recovery strategy for a Kafka mapping task" on page 30.

When you configure a mapping to read data from a Kafka topic in real-time, consider the following rule and guideline:

In the first mapping run, the mapping reads the messages in the Kafka topic based on the value you specify for the **Start Position Offset** source advanced property. In the subsequent mapping runs, the **Recovery Strategy** advanced session property overrides the **Start Position Offset** property and reads the messages from the last checkpoint.

Configure Informatica fixed partitions

When you read data from a Kafka source topic in real-time, you can configure Informatica fixed partitioning to optimize the mapping performance at run time.

When you configure Informatica fixed partitions on a Kafka source topic in a mapping, Kafka Connector treats each Informatica partition as an individual consumer within a topic and assigns a specific number of topic partitions to each Informatica partition.

Consider the number of messages to be processed in the mapping to determine an appropriate number of Informatica partitions for the mapping. You can specify up to 64 Informatica partitions.

Consider the following rules and guidelines when you configure Informatica partitions on a Kafka source topic in a mapping:

- You can only write data to relational targets that can process partitioned data.
- You cannot write data to a flat file target.
- After you run a mapping, you must not change the number of Informatica partitions. Otherwise, when you re-run the mapping, the mapping fails and the Secure Agent does not recover the messages in the Kafka topic.
- You must not configure Informatica partitions for a Kafka topic in an existing mapping that did not contain any partitions initially.

Kafka targets in mappings

To write data to a Kafka target, configure a Kafka object as the Target transformation in a mapping.

Specify the name and description of Kafka target. Configure the target and advanced properties for the target object in mappings.

The following table describes the target properties that you can configure for a Kafka target:

Property	Description
Connection	Name of the Kafka connection that is associated with a Kafka cluster.
Target Type	Type of the Kafka target objects available. You can write data to a single Kafka target object or parameterize the object. You cannot write data to multiple objects.
Object	Name of the Kafka target object based on the target type selected.
Parameter	Select a parameter for the target object, or click **New Parameter** to define a new parameter for the target object. The **Parameter** property appears only if you select Parameter as the target type.
Create New at Runtime[2]	Creates a target. Enter a name for the target object and path for the target object and select the source fields that you want to use. By default, all source fields are used. The target name can contain alphanumeric characters. You cannot use special characters in the file name except the underscore character (_). When you write data to a field of hierarchical data type, you cannot parameterize the target at runtime.
Format	The file format to write data to a Kafka topic. You can select from the following file format types: - None. Writes data to the Kafka topics in binary format. - Avro. Writes data to the Kafka topics in Avro format that use the binary encoding type. - JSON. Writes data to the Kafka topics in JSON format. To configure the formatting options for the file, click **Formatting Options**. For more information about format options, see "Formatting options for Kafka topics" on page 28.
Operation	Select **Insert** as the target operation. You cannot perform update, upsert, or delete operations on a Kafka target topic.
Preview Data	Preview the data of the Kafka target object. When you preview data, Data Integration displays the first 10 rows. By default, Data Integration displays the fields in native order. To display the fields in alphabetical order, enable the **Display source fields in alphabetical order** option.

[1] Applies only to mappings.

[2] Applies only to mappings in advanced mode.

The following table describes the advanced properties that you can configure for a Kafka target:

Property	Description
Metadata Fetch Timeout in milliseconds	The time after which the metadata is not fetched. Default value is 10000.
Batch Flush Time in milliseconds	The interval after which the data is published to the target. Default value is 1000.
Batch Flush Size in bytes	The batch size of the events after which the Secure Agent writes data to the target. Default value is 16384.
Producer Configuration Properties	The configuration properties for the producer. Overrides the **Additional Connection Properties** or **Additional Security Properties** specified in the Kafka connection. For more information about Kafka producer configuration properties, see Kafka documentation.
Forward Rejected Rows	Not applicable for Kafka Connector.

Edit metadata for the data field in a Kafka topic

You can edit the metadata for the data field in a Kafka topic to change the field of a binary data type to string data type.

When you read data from a Kafka topic in binary format and write the data to a target in string format, edit the metadata of the **data** field in the Kafka source and change the platform data type from binary to string.

When you read data from a source in string format and write the data to the **data** field in a Kafka topic in string format, edit the metadata of the **data** field in the Kafka target and change the platform data type from binary to string.

In both the scenarios, you need to change only the platform data type from binary to string without altering the native data type.

Formatting options for Kafka topics

When you select a Kafka source or a Kafka target topic, you can configure format options.

Schema Source

You must specify the schema of the source or target file. You can select one of the following options to specify a schema:

- **Import from Schema File**. Imports schema from a schema definition file in your local machine.
- **Read from data file**. Imports schema from a file in Kafka when you configure a Kafka connection to use the Confluent schema registry to import metadata.

Schema File

When you select the **Import Schema from File** option in the **Schema Source** drop-down list, you must choose a schema definition file on your local machine. You cannot upload a schema file when you select the **Read from data file** option.

Optionally, click **Show Data Preview** to preview data.

Sample schema files

The Kafka sources and targets in a mapping read or write data in JSON or Avro format. The following examples contain samples for each schema format.

Sample JSON File

When you use a Kafka source or target in a mapping, specify the format in which Kafka Connector reads or writes data. When you specify JSON format, provide a sample JSON file. The sample file contains dummy JSON data and is used to generate the schema.

The following is a sample JSON file:

```
{"id" : 1, "name" : "sample"}
```

Note: You must define the JSON sample in a single line. Otherwise, the mapping fails.

Sample Avro Schema

When you use a Kafka source or target in a mapping, specify the format in which Kafka Connector reads or writes data. When you specify Avro format, provide a sample Avro file.

The following schema is a sample Avro schema:

```
{
  "type" : "record",
  "name" : "Customer",
  "fields" : [ {
    "name" : "c_custkey",
    "type" : [ "int", "null" ],
    "default" : null
  }, {
    "name" : "c_name",
    "type" : [ "string", "null" ],
    "default" : null
  }, {
    "name" : "c_address",
    "type" : [ "string", "null" ],
    "default" : null
  }]
}
```

Hierarchy Builder transformation in Kafka mappings

When you read data from a relational source and write JSON data to a Kafka target, you must use a Hierarchy Builder transformation.

The transformation processes relational input from the upstream transformation and provides JSON output to the downstream transformation. The Hierarchy Builder transformation produces JSON output based on the

sample schema of the target table that you associate with the transformation and the way that you map the data.

For more information on using the Hierarchy Builder transformation with Kafka Connector, see https://docs.informatica.com/integration-cloud/cloud-data-integration-connectors/h2l/1544-converting-relational-input-into-hierarchical-output-using-.html.

Configure message recovery strategy for a Kafka mapping task

When you configure message recovery for a Kafka mapping task, the Secure Agent can recover unprocessed messages from a failed mapping. When you enable message recovery for a Kafka mapping task, the Secure Agent stores source messages or message IDs in a recovery file. If the mapping task fails, run the mapping task in recovery mode to recover the messages that the Secure Agent did not process. For mappings with Kafka topics, the Secure Agent connects to the recovery topic as a durable subscriber so messages persist even if the mapping task ends.

You can configure message recovery for tasks that read from a Kafka source and writes to a Kafka, Oracle, IBM MQ, or Flat File target.

When you configure a mapping task, you can configure the **Recovery Strategy** property in the **Advanced Session Properties** section on the **Runtime Options** tab of the mapping task and select **Resume from the last checkpoint**. The Secure Agent saves the mapping state of operation and maintains target recovery tables. If the mapping aborts, stops, or terminates, the Secure Agent uses the saved recovery information to resume the mapping from the point of interruption.

Steps to Enable Message Recovery

Complete the following steps to enable message recovery for a Kafka mapping task:

1. In the **Advanced Session Properties** on the **Runtime Options** tab of the mapping task, add the following properties:

Session Property Name	Session Property Value
Commit on End of File	No
Commit Type	Source
Recovery Strategy	Resume from last checkpoint

2. Click **Finish**.

The Secure Agent stores the messages in the recovery file and a backup of the recovery file in the following directory:

```
<Informatica Cloud Secure Agent installation directory>\apps\Data_Integration_Server\data
```

Note: The Secure Agent uses the following format for the recovery file name:

```
pmgmd_cciFile_<InternalSessionDetails><session_ID><partition_ID>
```

For example, the Secure Agent creates the recovery file with the following file name:

```
pmgmd_cciFile_00100s_mtt_0100CO0Z0000000000290300
```

Configure incremental load to read from Kafka topics

You can configure incremental load in a mapping in advanced mode to read from Kafka source topics.

When you re-run mappings enabled for incremental load, the mapping does not read records from previous runs.

To enable incremental load, configure the consumer property group ID in the **Consumer Configuration Properties** source advanced property. The group ID consumer property is the name of the consumer group to which the Kafka consumer belongs.

Enter the group ID of the consumer group in the `key1=value1` format.

For example, enter the following group ID of the consumer group: `group.id=consumer-1`.

To configure multiple consumer properties, separate each key-value pair with a comma.

When you configure the consumer property group ID, ensure that you set the **Start Position Offset** property to **Earliest** and the **Connector Mode** property to **Batch**.

The offset of the next record for the configured consumer group is stored after the successful completion of each mapping run. When you re-run the mapping, it starts reading data from the stored offset point.

If you set the **Start Position Offset** property to **Custom** and configure the **Custom Start Position Timestamp** property, the mapping ignores the recorded offset value saved from the previous run and starts reading data from the value set in the **Custom Start Position Timestamp** property. This option helps you customize exactly where in the Kafka log the mapping must start reading the data.

If you do not configure the group ID consumer property, the mapping automatically configures a unique consumer group name and does not incrementally load the data. Each mapping re-run reads all existing records in the source topic.

Rules and guidelines for Kafka mappings

Use the following rules and guidelines when you configure a Kafka mapping:

- You cannot parameterize the source object in a mapping.
- You cannot configure partitioning in a mapping that reads data from a Kafka source in batch mode.
- You cannot configure Lookup and Sorter transformations in a mapping.
- When you use a Secure Agent group to run a mapping, you cannot configure message recovery.
- When you configure a mapping that reads hierarchical data types from the source, the mapping might fail at runtime.
- When you select JSON as the format type, you cannot read hierarchical data from a Kafka source or write hierarchical data to a Kafka target.
- When you abort a mapping task or terminate the Data Transformation Manager (DTM) process, the number of success rows in the session log and the target results page of the mapping task do not match.
- For a Kafka connection used in an existing mapping that reads Avro data from or writes Avro data to a Kafka topic, do not configure the **Schema Registry URL** connection property. Otherwise, when you rerun the same mapping rerun, the mapping fails with the following error:
  `[ERROR] java.lang.RuntimeException: Failed : HTTP error code : 422`
- When you run a mapping that reads from a Kafka source and writes to a target of format JSON type and the target includes the key field, the mapping fails.

- The data preview in the Target transformation displays additional fields such as offset and header received from the source data. But these fields are not available in the Kafka target. You cannot map these fields to the target.
- You cannot configure a mapping to read data in Realtime mode if the Kafka connection is configured to use the Confluent schema registry.
- You can use the Avro formatting option with the Kafka connector that uses the Confluent schema registry.
- When you read Avro data from or write Avro data to a Kafka topic with Confluent Schema Registry, the mapping runs successfully with the latest registered schema for the selected topic. In the subsequent runs, if the schema is updated, ensure that you refresh the metadata for the mapping or create a new mapping.
- When you read from or write to a column of Boolean data type in a Kafka topic with Confluent schema registry, ensure that you specify 0 for False and 1 for True.
- When you read Avro data from or write Avro data to a Kafka topic with Confluent schema registry, you cannot view the schema in the formatting options.

Rules and guidelines for mappings in advanced mode

Consider the following guidelines when you create a mapping in advanced mode:

- You cannot configure a mapping to read data in Realtime mode.
- When you create a mapping in advanced mode, you cannot preview data for individual transformations to test the mapping logic.
- You cannot configure a Lookup transformation in a mapping in advanced mode.
- When you configure one-way or two-way SSL authentication to connect to a Kafka broker, ensure that the truststore or keystore file name doesn't contain spaces or UTF-8 characters.
- When you write data into an existing Kafka target in Avro format and import the schema with an Avro schema file, the Secure Agent ignores the schema in the Avro schema file and uses an automatically generated Avro schema to write data into the Kafka target.
- When you use the Confluent schema registry to import Avro metadata, you cannot write data into an existing Kafka target. To write data to a Kafka topic in Avro format, create a target runtime.
- You cannot use parameterized sources when you select the discover structure format.
- When you use an intelligent structure model with an Avro schema file and the Avro input file doesn't match the Avro schema file or only partially matches the Avro schema file, there might be data loss.
- When you use an intelligent structure model with an Avro schema file and the input file contains more columns than the Avro schema file, Intelligent Structure Discovery doesn't assign the data to an **Unassigned Data** field.
- When you use an intelligent structure model with an Avro schema file and there is a data type mismatch in the input file and schema file, Intelligent Structure Discovery doesn't assign the data to an **Unassigned Data** field.
- When you use an intelligent structure model in a source, ensure that you do not use transformations that do not support hierarchical data types. Otherwise, the mapping fails with the following error:
  `The transformation does not support fields that contain hierarchical data.`
- When you select Avro as the format type, you cannot preview data.

- When you select Avro as the format type for an existing target and configure a schema with columns of primitive data types, the default schema contains a field of Union data type with a primitive data type and a Null data type.
- When you import a Kafka target, ensure that there are no hierarchical fields in the target. To write data to a hierarchical field, create a target at runtime.
- When you monitor a mapping run in advanced mode, the **My Jobs** page does not display the number of rows that the Spark job processed.
- When you read Boolean data type in JSON format from an Amazon S3 source and write to a Kafka target, the data is written as an Integer data type.
- When you parameterize the **Custom Start Position Timestamp** source advanced property with an incorrect parameterization format like `$$$<parameter_name>`, the mapping fails with an irrelevant error in place of parameterization format error:
  `The Custom Start Position Timestamp cannot be empty when Start position offset is selected as [Custom] in the source data object [Source].`
- When you parameterize the **Custom Start Position Timestamp** source advanced property with an incorrect parameterization format like `$<parameter_name>`, the mapping doesn't fail.
- If the JSON data that you read from a source fails to align with the source schema defined in the schema definition file, the data written to the target appears corrupted.

APPENDIX A

Data type reference

Data Integration uses the following data types in mappings and mapping tasks with Kafka:

Kafka native data types

Kafka data types appear in the **Fields** tab for Source and Target transformations when you choose to edit metadata for the fields.

Transformation data types

Set of data types that appear in the transformations. They are internal data types based on ANSI SQL-92 generic data types, which the Secure Agent uses to move data across platforms. Transformation data types appear in all transformations in a mapping.

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

Kafka and transformation data types

When the Secure Agent reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When the Secure Agent writes to a target, it converts the transformation data types to the comparable native data types.

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

Kafka Data Type	Transformation Data Type	Range and Description
BINARY	Binary	1 to 104,857,600 bytes
INTEGER	Integer	Precision 10, scale 0
RAW	Binary	1 to 104,857,600 bytes
STRING	Nstring	1 to 104,857,600 characters
TIMESTAMP	Date/Time	Date and time values. (precision to the nanosecond)

Avro data types and transformation data types

Avro data types map to transformation data types that the Secure Agent uses to move data across platforms.

The following table lists the Avro data types that Data Integration supports for Kafka source and targets and the corresponding transformation data types:

Avro Kafka Data Type	Transformation Data Type	Range and Description
Boolean	Integer	TRUE (1) or FALSE (0)
Double	Double	Precision 15
Float	Double	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[1]	Map	Unlimited number of characters
Record[1]	Struct	Unlimited number of characters
String	String	1 to 104,857,600 characters Default precision is 256. You can increase the value up to 104857600 characters.
Union[1]	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.
[1]Does not apply when you use the Confluent schema registry to import Avro metadata.		

JSON data types and transformation data types

JSON data types map to transformation data types that the Secure Agent uses to move data across platforms.

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

JSON Kafka Data Type	Transformation Data Type	Range and Description
Array	Array	Unlimited number of characters
Boolean	Integer	TRUE (1) or FALSE (0)

JSON Kafka Data Type	Transformation Data Type	Range and Description
Double	Double	Precision 15
Integer	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0
Long	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision of 19 digits, scale of 0
Object	Struct	Unlimited number of characters
String	String	1 to 104,857,600 characters Default precision is 256. You can increase the value up to 104857600 characters.

INDEX