Informatica

Informatica® MDM - Relate 360

10.1

Installation and Configuration Guide

Informatica MDM - Relate 360 Installation and Configuration Guide
10.1
June 2019

Publication Date: 2019-06-27

Table of Contents

Preface

The *Informatica MDM - Relate 360 Installation and Configuration Guide* provides information about how to install and configure Relate 360 for system administrators and data stewards. This guide assumes that you are familiar with the interface requirements for the Hadoop environment.

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 Network

The Informatica Network is the gateway to many resources, including the Informatica Knowledge Base and Informatica Global Customer Support. To enter the Informatica Network, visit https://network.informatica.com.

As an Informatica Network member, you have the following options:

- Search the Knowledge Base for product resources.
- View product availability information.
- Create and 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 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 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.

Informatica maintains documentation for many products on the Informatica Knowledge Base in addition to the Documentation Portal. If you cannot find documentation for your product or product version on the Documentation Portal, search the Knowledge Base at https://search.informatica.com.

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

Informatica Product Availability Matrices

Product Availability Matrices (PAMs) indicate the versions of the operating systems, databases, and types of data sources and targets that a product release supports. You can browse the Informatica 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 and based on real-world experiences from hundreds of data management projects. Informatica Velocity represents the collective knowledge of Informatica consultants who work with organizations around the world to plan, develop, deploy, and maintain successful data management solutions.

You can find 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 extend and enhance your Informatica implementations. Leverage any of the hundreds of solutions from Informatica developers and partners on the Marketplace to improve your productivity and speed up time to implementation on your projects. You can find the Informatica Marketplace at https://marketplace.informatica.com.

Informatica Global Customer Support

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

To find your local Informatica Global Customer Support telephone number, visit the Informatica website at the following link:
https://www.informatica.com/services-and-training/customer-success-services/contact-us.html.

To find online support resources on the Informatica Network, visit https://network.informatica.com and select the eSupport option.

CHAPTER 1

Installing Informatica MDM - Relate 360

This chapter includes the following topics:

Installation Overview

The Informatica MDM - Relate 360 installer is distributed as a RedHat Package Manager (RPM) installation package.

The RPM package includes batch jobs, utilities to run search service, and enablers for data warehouses.

Pre-Installation Tasks

Before you begin the installation, perform the following pre-installation tasks:

- Verify that the following softwares are installed:
    - A Hadoop distribution with HDFS and MapReduce
    - HBase if you want to persist the linked data in a repository
    - Apache ZooKeeper
    - Apache Tomcat
    - Apache Spark if you stream input data

- Apache Kafka if you stream input data
    - Hive data warehouse if you want to perform analytics in a Hive environment
    - Java Developer Kit (JDK) and Java Runtime Environment (JRE)
- Verify that the administrator has the root privileges.
- Copy the following package to a temporary directory: `MDMRelate360-<Version Number>.x86_64.rpm`

For more information about product requirements and supported platforms, see the Product Availability Matrix on Informatica Network:
https://network.informatica.com/community/informatica-network/product-availability-matrices

Installing Relate 360

Relate 360 installation includes installing the RPM package and adding the required population files to the installation directory. A population file contains rules specific to a particular population of data. The rules define how to build keys and how the search and match strategies function for the specified population.

1. Install Relate 360.
2. Add the population files to the installation directory.

Step 1. Install Relate 360

You must have the root privileges to install Relate 360.

From the root directory, run the following command:

```
rpm -ivh <Absolute Path>/MDMRelate360-<Version Number>.x86_64.rpm
```

`Version Number` indicates the version number of the Relate 360, and `Absolute Path` indicates the absolute path to the RPM package.

For example, `rpm -ivh /usr/local/MDMRelate360-10.1-0.x86_64.rpm`

The installer installs Relate 360 in the following directory: `/usr/local/mdmbdrm-<Version Number>`

Step 2. Add the Population Files to the Installation Directory

After you install Relate 360, contact Informatica Global Customer Support for the required population files and add the population files to the Relate 360 installation directory.

To add the population files to the installation directory, copy the population files to the following directory: `/usr/local/mdmbdrm-<Version Number>/population`

Granting User Permissions for the Amazon EMR Environment

After you install Relate 360 in the Amazon EMR environment, you can use the default user name, `hadoop`, or other user names to access the Relate 360 batch jobs.

If you use `hadoop` as the user name, you do not require any additional permissions to run the batch jobs. If you use other user names, ensure that you grant the write permission to the working directory in HDFS and the read permission to the input data in HDFS before you run the batch jobs.

Coping the Hive JDBC Driver

If you use a Secure Sockets Layer (SSL) enabled Hive server, you must copy the Hive JDBC driver to the Relate 360 installation directory.

1. Based on the Hadoop distribution that you use, copy the `hive-jdbc-<version>-standalone.jar` to the following directory:
   `/usr/local/mdmbdrm-10.1/bin`.
2. Rename the copied file to `hive-jdbc.jar`.
3. Set the following permission for the file:
   `chmod 777 hive-jdbc.jar`.

Upgrading Relate 360

When you upgrade a Relate 360 installation, the installer upgrades all the binary files but does not remove any custom files that you add to the installation directory, such as the population files. You must have root privileges to perform the upgrade process.

1. If you use streaming data, kill the Spark application that you use to process the streaming data.
2. Back up the installation directory of Relate 360. The default installation directory is `/usr/local/mdmbdrm-<Version Number>`
3. From the root directory, run the following command:

   ```
   rpm -ivh <Local Directory>/MDMRelate360-<Version Number>.x86_64.rpm
   ```

   `Version Number` indicates the version number of the Relate 360, and `Absolute Path` indicates the absolute path to the RPM package.

   For example, `rpm -ivh MDMRelate360-10.1-0.x86_64.rpm`

   The installer upgrades all the binary files in the following directory: `/usr/local/mdmbdrm-<Version Number>`

   Note: Do not use the `-uvh` option to upgrade a Relate 360 installation.

After You Upgrade

If you plan to consolidate the linked data and create preferred records, you must create an empty preferred records table in the repository. Use the `create_preferred_records_table.sh` script located in the following directory to create an empty preferred records table: `/usr/local/mdmbdrm-<Version Number>`

1. Use the following command to run the `create_preferred_records_table.sh` script:

   ```
   create_preferred_records_table.sh --config=<Configuration file name>
   ```

   The `Configuration file name` parameter indicates the absolute path and file name of the configuration file that you plan to use.

   The following sample command runs the `create_preferred_records_table.sh` script:

   ```
   create_preferred_records_table.sh --config=/usr/local/conf/config_big.xml
   ```
2. To create preferred records for all the clusters in the preferred records table, run the consolidation job in the regenerate mode.
3. If you use Spark to process streaming data, run the `setup_realtime.sh` script located in the following directory to redeploy Relate 360 on Spark: `/usr/local/mdmbdrm-<Version Number>`

   Note: Ensure that you specify a different checkpoint directory and `--consolidate` option when you run the `setup_realtime.sh` script.

RELATED TOPICS:

- "Creating a Configuration File" on page 11
- "Step 3. Deploy Relate 360 on Spark" on page 67

Uninstalling Relate 360

When you uninstall Relate 360, the uninstaller deletes all the binary files but does not delete the files that you manually add, such as the population files. You must have the root privileges to uninstall a Relate 360 installation.

From the root directory, run the following command:

```
rpm -e <Installed Package Name>
```

For example, `rpm -e MDMRelate360-10.0.HF9-0.x86_64`

The uninstaller deletes all the binary files but does not delete the folders located in the following directory: `/usr/local/mdmbdrm-<Version Number>`

CHAPTER 2

Configuring Relate 360

This chapter includes the following topics:

Configuration Overview

After you install Informatica MDM - Relate 360, you must create a configuration file. A configuration file includes parameters related to the Hadoop distribution, repository, input record layout, and metadata.

You can create a matching rules file based on your requirement. A matching rules file includes parameters related to matching rules. The behavior of the batch jobs depends on the parameters that you configure in the configuration file and the matching rules file.

If you plan to consolidate the linked data, you must create a consolidation rules file. The consolidation rules file contains the row, column, and default rules that the consolidation process uses to create a preferred record for each cluster.

If you plan to create relationships between the processed data, you must create a relationship configuration file. A relationship configuration file defines the business entity types and their potential relationships. If you plan to view the relationship graph that you create, you must create a properties file for the relationship graph user interface.

Creating a Configuration File

You must create a configuration file in the XML format to include parameters related to the Hadoop distribution, repository, input record layout, metadata, and Hive.

You can customize the following sample configuration file based on your environment and requirement: `/usr/local/mdmbdrm-<Version Number>/sample/MDMBDRMTemplateConfiguration.xml`

Configuring the Hadoop Distribution Parameters

You can configure parameters related to the Hadoop distribution, such as split size, in the configuration file.

The number of MapReduce jobs that you want to process the input file depends on the split size. The longer the block size, the longer run time for a single job. If you split the input file into multiple parts based on the split size, a separate job processes each part, which improves the run time.

For example, the input file size is 112 MB. If the block size is 128 MB, the HDFS stores the input file in a single block, and a single job processes the file. If you set the split size as 32 MB, the input file is split into 4 parts and 4 jobs process the file.

To configure the Hadoop distribution, add the following parameters to the `HadoopConfiguration` section in the configuration file:

JobName

Optional. Name for the configuration that you create.

MinInputSplitSize

Optional. Minimum valid size in bytes to split a file. Default is 0.

Note: The `MinInputSplitSize` parameter overrides the `mapred.min.split.size` property of Hadoop when you run a job.

MaxInputSplitSize

Optional. Maximum valid size in bytes to split a file.

By default, the split size is equal to the HDFS block size.

Note: The `MaxInputSplitSize` parameter overrides the `mapred.max.split.size` property of Hadoop when you run a job.

The following sample code shows the parameters for the Hadoop distribution:

```
<HadoopConfiguration>
     <JobName>Extract Job</JobName>
     <MinInputSplitSize>0</MinInputSplitSize>
     <MaxInputSplitSize>33554432</MaxInputSplitSize>
</HadoopConfiguration>
```

Configuring the Repository Parameters

You must configure parameters related to the repository, such as the host server name and the port number on which the server listens, in the configuration file.

To configure the repository parameters, add the following parameters to the `HBASEConfiguration` section in the configuration file:

HbaseMaster

Host name of the HBase Master server and the port number on which the Master server listens.

Specify the port number based on the `hbase.master.port` parameter configured in HBase.

Use the following format to specify a value for the `HbaseMaster` parameter:

```
<Master Server Host Name>:<Port>
```

HbaseZookeeperQuorum

Comma-separated list of ZooKeeper servers when HBase uses an ensemble of ZooKeeper servers.

For example: server1.domain.com,server3.domain.com

You can get the list of servers from the `hbase.zookeeper.quorum` property in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseZookeeperClientPort

Optional. Port number on which the ZooKeeper server listens for client connections. Default is 2181.

HbaseRootDirectory

Required if you use Hortonworks Data Platform. Directory that the region servers share on the local file system. Default is `${hbase.tmp.dir}/local`.

Note: The `HbaseRootDirectory` parameter overrides the `hbase.rootdir` parameter configured in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseDistributed

Optional. Indicates whether the HBase runs in the standalone or distributed mode.

Set to true to indicate distributed mode and set to false to indicate standalone mode. In the standalone mode, HBase runs all the HBase and ZooKeeper daemons in a single Java virtual machine (JVM). Default is false.

Note: The `HbaseDistributed` parameter overrides the `hbase.cluster.distributed` parameter configured in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseZookeeperZnodeParent

Required if you use Hortonworks Data Platform. Root ZNode path for the ZooKeeper files. HBase stores all the ZooKeeper files configured with the relative path in the root ZNode path. Default is `/hbase`.

Note: The `HbaseZookeeperZnodeParent` parameter overrides the `zookeeper.znode.parent` parameter configured in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseCompressionAlgorithm

Optional. Compression algorithm that you want HBase to use. Use one of the following compression algorithms:

- SNAPPY
- LZO
- LZ4
- GZ
- NONE

Default is NONE.

HbaseDataBlockEncoding

Optional. Type of data block encoding that you want HBase to use. Use one of the following data block encoding types:

- PREFIX
- DIFF
- FAST_DIFF
- NONE

Default is NONE.

ScanBatchSize

Optional. Number of records to return on each scan. Default is 100.

ScanCacheSize

Optional. Number of records to pass to scanners at once. Adjust the value of the `ScanCacheSize` parameter based on the `ScanMaxResultSize` parameter.

A lower value can result in multiple scans that might affect the performance of the jobs. A higher value can result in the failure of jobs if the size of the records exceeds the value of the `ScanMaxResultSize` parameter.

Default is 500.

ScanMaxResultSize

Optional. Maximum size of records in bytes to return to scanners at once. Adjust the value of the `ScanMaxResultSize` parameter based on the `ScanCacheSize` parameter.

A lower value can result in multiple scans and a higher value can result in increased memory usage that might affect the performance of the jobs.

Default is 2097152.

CacheBlock

Optional. Indicates whether you want to enable block cache for the scan.

Set to true to enable block cache and set to false to disable block cache. Default is false.

AutoFlush

Optional. Indicates whether you want to enable auto flush behavior.

Set to true if you want to enable auto flush and set to false if you want to disable auto flush. Default is false.

WALonPUT

Optional. Indicates whether you want to enable Write Ahead Log (WAL) edits for a put method.

Set to true to enable WAL edits for a put method and set to false to disable WAL edits for the put method. When you set to false, the region server does not write the logs to the file-based storage and your data might be at risk. Default is false.

EnableSmallScan

Optional. Indicates whether you want to enable small scan.

Set to true if you want to enable small scan and set to false if you want to disable small scan. Default is false.

RegionSplitSize

Optional. Key size to split the region. The value enables region split policy and groups records based on the prefix of the row key. Default is 8 bytes.

DriverName

Optional. Name of the HBase driver to use. The supported driver name is `com.informatica.mdmbde.database.hbase.HBaseDatabaseAdapterImplVOne`.

CoprocessorPath

Absolute path and file name for the coprocessor JAR file that you must generate and deploy in HDFS. The JAR file contains the search logic that the region servers use to perform searches.

For example, `/user/cloudera/db-hbase-coprocessorDeploy.jar`.

Use the following name format for the JAR file name: `db-hbase-coprocessor<id>`

The name format uses the `id` parameter that indicates a unique identifier for the JAR file. For example, if `id=Deploy`, the JAR file name must be `db-hbase-coprocessorDeploy.jar`.

CoprocessorClass

Name of the class that the coprocessor uses. Specify `com.informatica.mdmbde.database.hbase.coprocessor.BDRMRegionObserver` as the parameter value.

SearchTokenValidity

Optional. Number of seconds that a search token remains valid. When you enable pagination, a search request returns a token with the search results. You can use the token to get the subsequent pages of the search results from cache to avoid performing the search again. The token expires after the specified time. Default is 600 seconds.

KeyTabFile

Optional. Absolute path and file name of the keytab file. A keytab file contains a list of keys that are analogous to user passwords. Applicable if you use Kerberos for authentication.

Note: If you use the `KeyTabFile` parameter, ensure that the name of the file and the absolute path to the file are the same for all the nodes in a distributed Hadoop cluster.

PrincipalName

Required if you use Kerberos for authentication. Service Principal Name (SPN) of the HBase master server. For example, hbase/_Host@realm.com.

You can get the SPN of the HBase master server from the `hbase.master.kerberos.principal` property in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

The following sample code shows the parameters for HBase:

```
<HBASEConfiguration>
   <HbaseMaster>HadoopServer:60000</HbaseMaster>
   <HbaseZookeeperClientPort>2181</HbaseZookeeperClientPort>
   <HbaseZookeeperQuorum>iir-hadoop-test1</HbaseZookeeperQuorum>
   <HbaseRootDirectory />
   <HbaseDistributed>true</HbaseDistributed>
   <HbaseZookeeperZnodeParent />
   <HbaseCompressionAlgorithm>SNAPPY</HbaseCompressionAlgorithm>
   <HbaseDataBlockEncoding>PREFIX</HbaseDataBlockEncoding>
   <ScanCacheSize>100000</ScanCacheSize>
   <ScanMaxResultSize>2097152</ScanMaxResultSize>
   <CacheBlock>false</CacheBlock>
   <AutoFlush>false</AutoFlush>
   <WALonPUT>false</WALonPUT>
   <ScanBatchSize>100</ScanBatchSize>
   <EnableSmallScan>false</EnableSmallScan>
   <RegionSplitSize>8</RegionSplitSize>
   <DriverName>com.informatica.mdmbde.database.hbase.HBaseDatabaseAdapterImplVOne</
DriverName>
   <SearchTokenValidity>1000</SearchTokenValidity>
   <KeyTabFile>/etc/security/keytabs/hbase.keytab</KeyTabFile>
   <PrincipalName>hbase/_Host@realm.com</PrincipalName>
</HBASEConfiguration>
```

Configuring the SSA-NAME3 Parameters

SSA-NAME3 uses population files that contain rules specific to the particular population of data. The rules define how to build keys and how the search and match strategies function for the specified population. SSA-

NAME3 builds keys, generates an array of search key ranges, and uses the key ranges to identify records for matching.

SSA-NAME3 uses match purposes to match different types of data. A match purpose is a predefined algorithm that is optimized for a specific type of data, such as name or address. The algorithms include numerous rules that handle initials, aliases, common variations, prefixes, suffixes, transpositions, and word order. Each match purpose contains a group of SSA-NAME3 fields, and each field adds weight to the matching score.

To configure the SSA-NAME3 parameters, perform the following tasks:

1. Configure the SSA-NAME3 properties.
2. Configure the SSA-NAME3 index parameters.
3. Configure the SSA-NAME3 search parameters.
4. Configure the SSA-NAME3 match parameters.

Configuring the SSA-NAME3 Properties

You must specify the SSA-NAME3 properties, such as the population file that you want to use, in the configuration file.

To configure the SSA-NAME3 properties, add the following parameters to the `nm3configuration` section in the configuration file:

ssapr

Absolute path to the SSA-NAME3 population files.

You can find the population files in the following directory: `/usr/local/mdmbdrm-<Version Number>/population`

ssabin

Absolute path to the SSA-NAME3 library files.

You can find the SSA-NAME3 library files in the following directory: `/usr/local/mdmbdrm-<Version Number>/bin`

population

Name of the population file that you want to use.

keysize

Optional. Size of the keys that SSA-NAME3 create. Use one of the following sizes:

- 5 bytes
- 8 bytes

Default is 8 bytes.

keytype

Optional. Format of the SSA-NAME3 fuzzy keys. Use one of the following values:

- DEFAULT. Indicates that the keys are in the standard SSA-NAME3 format.
- HEX. Indicates that keys are in the hexadecimal format.

The hexadecimal format results in improved performance than the standard SSA-NAME3 format. Default is DEFAULT.

The following sample code shows the SSA-NAME3 properties:

```
<nm3configuration>
      <ssapr>/usr/local/mdmbdrm-<Version Number>/population</ssapr>
```

```
        <ssabin>/usr/local/mdmbdrm-<Version Number>/bin</ssabin>
        <population>usa</population>
        <keysize>8</keysize>
        <keytype>HEX</keytype>
</nm3configuration>
```

Configuring the SSA-NAME3 Index Parameters

You must configure the SSA-NAME3 index parameters, such as the column name that you want to use to index the records, in the configuration file.

To configure the SSA-NAME3 indexing parameters, add the following parameters to the `IndexingConfiguration` section in the configuration file:

indexFieldName

Name of the column that you want to use to index the records.

Note: Ensure that you specify the column name in the `PZMAP` section.

keyField

Name of the SSA-NAME3 field based on which you want to build keys.

Note: You can specify only one field as a key field in a configuration file. If you want to create indexes for other fields, you must create a separate configuration file for the required field.

keyLevel

Optional. Type of key level that you want to build. Use one of the following values:

- Standard. Builds more variations than limited key level but uses less disk space than extended key level.
- Extended. Builds more variations than standard key level and uses more disk space than standard and limited key levels.
- Limited. Builds less variations and uses low disk space than standard and extended key levels.

Default is Standard.

AdditionalControl

Optional. Additional attributes that you want to configure. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.
- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

The following sample code shows the SSA-NAME3 indexing parameters:

```
<IndexingConfiguration>
       <indexFieldName>NAME</indexFieldName>
       <keyField>Person_Name</keyField>
       <keyLevel>Standard</keyLevel>
       <AdditionalControl>NAMEFORMAT=L</AdditionalControl>
</IndexingConfiguration>
```

Configuring the SSA-NAME3 Search Parameters

You must configure the SSA-NAME3 search parameters, such as the field name based on which you want to search records, in the configuration file.

To configure the SSA-NAME3 searching parameters, add the following parameters to the `SearchConfiguration` section in the configuration file:

searchType

Name for the search that you configure.

searchFieldName

Name of the column based on which you want to generate key ranges to search records.

keyField

Name of the SSA-NAME3 field based on which you want to search records.

Note: You can specify only one field as a key field in a configuration file. If you want to use other fields, you must create a separate configuration file for the required field.

searchLevel

Required only if you use fuzzy keys for indexing. Type of search that you want to perform. Use one of the following values:

- Narrow
- Typical
- Exhaustive
- Extreme

A search returns more candidates as the search level increases but uses more resources and takes a longer execution time.

AdditionalControl

Optional. Additional attributes that you want to configure. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.
- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

The following sample code shows the SSA-NAME3 search parameters:

```
<SearchConfiguration>
      <searchType>Household</searchType>
      <searchFieldName>NAME</searchFieldName>
      <keyField>Person_Name</keyField>
      <searchLevel>Typical</searchLevel>
      <AdditionalControl>NAMEFORMAT=L</AdditionalControl>
</SearchConfiguration>
```

Configuring the SSA-NAME3 Match Parameters

You must configure the SSA-NAME3 match parameters, such as the SSA-NAME3 purpose that you want to use to match the records, in the configuration file.

To configure the SSA-NAME3 matching parameters, add the following parameters to the `MatchConfiguration` section in the configuration file:

Purpose

Type of purpose that you want to use for matching. You can use one of the following standard SSA-NAME3 purposes:

- Address. Identifies an address match.
- Contact. Identifies a contact within an organization at a specific location.
- Division. Identifies an organization at an address.

- Fields. Identifies generic data.
- Household. Identifies individuals with same or similar family names who share the same address.
- Individual. Identifies a specific individual by name, ID, or date of birth.
- Organization. Identifies an organization by name.
- Person_Name. Identifies a person by name.
- Resident. Identifies a person at an address.
- Wide_Contact. Identifies a contact within an organization.

MatchLevel

Optional. Level of matching that you want to perform. Use one of the following values:

- Typical. Returns more results that the conservative match level and less results than the loose match level.
- Conservative. Returns almost accurate results, and you can use in environments where the accuracy of a match is important.
- Loose. Returns matches with more variations than typical match, and you can use in environments where you can manually review the results.

Default is Typical.

Threshold

Minimum score that SSA-NAME3 requires to consider a record as a matching candidate.

MatchField

Maps the SSA-NAME3 fields with the input record fields. Use the following format to map the SSA-NAME3 and input record fields:

```
<MatchField>
   <MField name="Person_Name">NAME</MField>
   <MField name="Address_Part1">ADDRESS1</MField>
   <MField name="Address_Part2">ADDRESS2</MField>
</MatchField>
```

You can use the following standard SSA-NAME3 fields:

- Address_Part1. Indicates a part of address that includes building name, street number, street name, street type, and apartment details.
- Address_Part2. Indicates a part of address that includes city or town name, state name, ZIP code, and country.
- Attribute 1 and Attribute 2. Indicates generic data.
- Code. Indicates any numeric or alphanumeric codes, such as telephone number, license number, or credit card number.
- Company_Name. Indicates names of organizations, such as business names, institution names, department names, agency names, or trading names.
- CreditCard. Indicates credit card numbers.
- Date. Indicates dates, such as expiry date, date of contract, date of change, or creation date.
- Geocode. Indicates the latitude and longitude geographic coordinates with an optional elevation.
- ID. Indicates any type of ID numbers, such as account numbers, customer numbers, credit card numbers, drivers license numbers, passports, policy numbers, SSNs, or VINs.

- ISBN10 and ISBN13. Indicates International Standard Book Number (ISBN). Use the ISBN10 field for a 10-digit number, and use the ISBN13 field for a 13-digit number.
- Organization_Name. Indicates names of organizations, such as business names, institution names, department names, agency names, or trading names.
- Person_Name. Indicates names of people.
- Postal_Area. Indicates ZIP codes of towns or cities.
- Telephone_Number. Indicates telephone numbers.
- VIN. Indicates Vehicle Identification Numbers (VINs).

AdditionalControl

Optional. Additional attributes that you want to specify. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.
- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

MaxCandidateSet

Optional. Maximum number of records that can be processed for matching. By default, SSA-NAME3 processes all the records.

For example, if you set the maximum candidate size to 500, SSA-NAME3 uses the first 500 records for matching.

Note: If you want to process the streaming data, do not configure the `MaxCandidateSet` parameter. The `MaxCandidateSet` parameter value impacts the maximum number of records that the linking process can add to a cluster.

The following sample code shows the SSA-NAME3 match parameters:

```
<MatchConfiguration>
      <Purpose>Person_Name</Purpose>
      <MatchLevel>Typical</MatchLevel>
      <Threshold>80</Threshold>
      <MatchField>
            <MField name="Person_Name">NAME</MField>
            <MField name="Address_Part1">ADDRESS1</MField>
            <MField name="Address_Part2">ADDRESS2</MField>
      </MatchField>
      <MaxCandidateSet>500</MaxCandidateSet>
      <AdditionalControl>NAMEFORMAT=R</AdditionalControl>
</MatchConfiguration>
```

Configuring the Input Record Layout

You must define the column names and widths for the fixed-width input data in the `PZMAP` section in the configuration file.

The following sample code shows column names and widths for the input data:

```
<PZMAP>
      <COLUMN name="ROWID" length="10"/>
      <COLUMN name="ADDRESS1" length="24"/>
      <COLUMN name="ADDRESS2" length="24"/>
      <COLUMN name="CITY" length="14"/>
      <COLUMN name="STATE" length="2"/>
      <COLUMN name="ZIP" length="5"/>
      <COLUMN name="NAME" length="24"/>
      <COLUMN name="AGE" length="3"/>
      <COLUMN name="INCOME" length="10"/>
</PZMAP>
```

Configuring Metadata

You must configure the metadata information, such as the name of the column that you want to set as primary key, in the configuration file.

To configure the metadata information, add the following parameters to the `MetaData` section in the configuration file:

PK

Name of the column that you want to set as primary key.

SOURCE_COLUMN_NAME

Name of the column to store the source information of the data.

Note: You can use only `LMT_SOURCE_NAME` as the column name.

You can also use the `part_of_layout` attribute to specify whether the source information is part of the input data. Set to YES if the source information is part of the input data, and set to NO if the source information is not part of the input data. If you set to NO, ensure that you specify the `SOURCE_NAME` parameter.

For example: `<SOURCE_COLUMN_NAME part_of_layout="YES">LMT_SOURCE_NAME</SOURCE_COLUMN_NAME>`

SOURCE_NAME

Name of the source for the input data. If the input data does not contain the source name, use the `SOURCE_NAME` parameter to specify the source name. The source name cannot exceed 32 bytes.

For example: `<SOURCE_NAME is_reference="YES">PRIZM</SOURCE_NAME>`

CLUSTER_COLUMN_NAME

Name of the column on which you want to store the link identifiers.

CLUSTER_COLUMN_SIZE

Size of the column that stores the link identifiers. Use 40 bytes as the column size.

CLUSTER_OPTION

Indicates whether you want to delete the tables that the load job created in the repository when you run the load job again.

Set to true if you want to delete the tables, and set to false if you want to append the data to the existing tables. Default is false.

MATCHSOURCES

Specifies the source of the data that you want the MapReduce jobs to use.

For example:

```
<MATCHSOURCES>
        <MATCHSOURCE>ORACLE</MATCHSOURCE>
        <MATCHSOURCE>MYSQL</MATCHSOURCE>
        <MATCHSOURCE>SAP</MATCHSOURCE>
 </MATCHSOURCES>
```

The previous example specifies to include data only from Oracle, MySQL, and SAP for the MapReduce jobs to process.

ALTERNATETABLEFORGROUPINFO

Optional. Indicates whether you want to have a separate table to store the link information.

Set to true if you want to have a separate table, and set to false if you do not want to have a separate table. If the value is false, ensure that you specify the `ADDGROUPNUMBERTOROWKEY` parameter. Default is false.

ADDGROUPNUMBERTOROWKEY

Indicates whether you want to add the link number to the record key.

If you set `ALTERNATETABLEFORGROUPINFO=false`, set `AddGroupNumberToRowKey=true`. Default is false.

DELETEBATCHSIZE

Optional. Indicates the total number of records that you can delete at once. Default is 1000.

PARTITION_COLUMN_NAME

Optional. Indicates the name of the column based on which you can create a partition identifier and add the partition identifier to the key. Use the following attributes to define the partition identifier:

- length. Defines the length of the column that you can add as the prefix to the keys. The maximum length of the column that you can use is 8 bytes. Default is 2 bytes.
- part_of_rowkey. Indicates whether the key includes the partition identifier. Set to true if you want to prefix the key with the partition identifier, and set to false if you do not want to prefix the key with the partition identifier.

For example:

```
<PARTITION_COLUMN_NAME length="2" part_of_rowkey="YES">STATE</PARTITION_COLUMN_NAME>
```

Note: If you configure the `PARTITION_COLUMN_NAME` parameter for the initial linking job, you must configure the `PARTITION_COLUMN_NAME` parameter when you run other jobs to update or increment the initial data.

LinkTableName

Base name for the tables that the initial loading job creates in the repository. The initial loading job uses the following format for the table names:

```
MDMBDRM<OrganizationID>_<LINKTABLENAME>_<PK|GROUP>
```

For example, if you specify `LINKTABLENAME=LMT_MATCHED`, the initial loading job creates an index table named `MDMBDRM<OrganizationID>_LMT_MATCHED`, a primary key table named `MDMBDRM<OrganizationID>_LMT_MATCHED_PK`, and a link table named `MDMBDRM<OrganizationID>_LMT_MATCHED_GROUP`.

StoreAllFields

Optional. Indicates whether to persist all the columns that you define in the `PZMAP` section in the repository.

Set to true if you want to persist all the columns in the repository. Set to false if you want to persist only the columns that you use to index data in the repository. Default is false.

Note: If you plan to run the initial linking, initial loading, incremental linking, update linking, or repository data deletion job with the matching rules file, you must set `StoreAllFields=true`.

ColumnFamilyName

Name of the column family that groups all the columns in the repository table.

MaxConcurrentSessions

Maximum number of REST requests that you can run concurrently. A higher number improves search performance but uses more memory. You can configure the value based on the amount of available memory. Default is 200.

The following sample code shows the metadata configuration:

```
<MetaData>
   <PK>ROWID</PK>
   <SOURCE_NAME is_reference="YES">PRIZM</SOURCE_NAME>
   <SOURCE_COLUMN_NAME part_of_layout="YES">LMT_SOURCE_NAME</SOURCE_COLUMN_NAME>
   <CLUSTER_COLUMN_NAME>GROUPNO</CLUSTER_COLUMN_NAME>
   <CLUSTER_COLUMN_SIZE>40</CLUSTER_COLUMN_SIZE>
   <CLUSTER_OPTION deleteLinkTable="TRUE" />
   <MATCHSOURCES>
      <MATCHSOURCE>ORACLE</MATCHSOURCE>
      <MATCHSOURCE>MYSQL</MATCHSOURCE>
      <MATCHSOURCE>SAP</MATCHSOURCE>
   </MATCHSOURCES>
   <ALTERNATETABLEFORGROUPINFO>false</ALTERNATETABLEFORGROUPINFO>
   <DELETEBATCHSIZE>1000</DELETEBATCHSIZE>
   <PARTITION_COLUMN_NAME length="2" part_of_rowkey="YES">STATE</PARTITION_COLUMN_NAME>
   <LinkTableName>MDM_INDIVIDUAL_LMT_GA</LinkTableName>
   <ColumnFamilyName>MDMBDE_link_columns</ColumnFamilyName>
   <StoreAllFields>true</StoreAllFields>
   <AddGroupNumberToRowKey>true</AddGroupNumberToRowKey>
   <MaxConcurrentSessions>100</MaxConcurrentSessions>
</MetaData>
```

Configuring the Kafka Parameters

You can configure parameters related to Kafka, such as broker details and topic for the input data.

To configure the Kafka parameters, add the following parameters to the `RealTimeService` section in the configuration file:

Broker

Host name of the machine that hosts Kafka and the port number on which Kafka listens. If you use a Kafka cluster, you can specify multiple host names and port numbers separated by commas.

Use the following format to specify the host name and the port number:

```
<Host Name1>:<Port1>,<Host Name2>:<Port2>,...<Host NameN>:<PortN>
```

For example,`localhost:9092,KafkaBroker1:9092`

TopicName

Name for the topic to which Kafka publishes the input data stream.

The following sample code shows the parameters for Kafka:

```
<RealTimeService>
   <Broker>localhost:9092</Broker>
   <TopicName>test-22</TopicName>
</RealTimeService>
```

Configuring the Hive Parameters

You can configure parameters related to Hive, such as the JDBC driver for Hive, Service Principal Name (SPN) of the Hive master server, and the keytab file.

To configure the Hive parameters, add the following parameters to the `HiveConfiguration` section in the configuration file:

KeyTabFile

Optional. Absolute path and file name of the keytab file. A keytab file contains a list of keys that are analogous to user passwords. Applicable if you use Kerberos for authentication.

Note: If you use the `KeyTabFile` parameter, ensure that the name of the file and the absolute path to the file are the same for all the nodes in a distributed Hadoop cluster.

PrincipalName

Required if you use Kerberos for authentication. Service Principal Name (SPN) of the Hive master server. For example, hive/_Host@realm.com.

You can get the SPN of the Hive master server from the `hive.metastore.kerberos.principal` property in the following file: `${HIVE_HOME}/conf/hive-site.xml`

JDBCUrl

JDBC connection URL to access metadata from Hive.

Use the following format for the JDBC connection URL:

```
jdbc:hive2://<Host Name>:<Port>/default;<Additional Parameter 1>;<Additional
Parameter 2>,...<Additional Parameter N>
```

The URL format uses the following parameters:

- `Host Name`. Name of the machine that hosts the Hive master server.
- `Port`. Port number on which the Hive master server listens.
- `Additional Parameter 1,2,...N`. Additional parameters that are specific to your environment. For example, in an encrypted environment, use the `sasl.qop` parameter to specify the type of authentication you use. For example, `sasl.qop=auth-conf` indicates that the environment uses authentication with confidentiality protection. For more information about the `sasl.qop` parameter, see the documentation of your Hadoop distribution.

The following sample code shows the Hive parameters for Cloudera CDH:

```
<HiveConfiguration>
   <JDBCUrl>jdbc:hive2://1.254.254.254:10000/default;ssl=true;sslTrustStore=/etc/
cloudera-scm-agent/custom.truststore;trustStorePassword=hadoop</JDBCUrl>
   <KeyTabFile>/run/cloudera-scm-agent/process/232-hive-HIVESERVER2/hive.keytab</
KeyTabFile>
   <PrincipalName>hive/host@principal</PrincipalName>
</HiveConfiguration>
```

The following sample code shows the Hive parameters for Hortonworks HDP:

```
<HiveConfiguration>
   <JDBCUrl>jdbc:hive2://1.253.253.253:10001/
default;transportMode=http;httpPath=cliservice;ssl=true;sslTrustStore=/etc/hive/conf/
hivetrust.jks;trustStorePassword=hadoop</JDBCUrl>
   <KeyTabFile>/run/hdp-scm-agent/process/232-hive-HIVESERVER2/hive.keytab</KeyTabFile>
   <PrincipalName>hive/host@principal</PrincipalName>
</HiveConfiguration>
```

Creating a Matching Rules File

You can create a matching rules file in the XML format to define indexes, searches, and matching rules for each index. The values that you configure in the matching rules file override the values in the configuration file.

You can customize the following sample matching rules file based on your requirement: `/usr/local/mdmbdrm-<Version Number>/sample/MDMBDRMMatchRuleTemplate.xml`

Configuring the Match Rule Set

You can define one or more match rule sets in a matching rules file. A match rule set definition includes an index, one or more searches, and one or more matching rules. You can define a match rule set within the `MDMBDRMMatchRulesSet` section.

To define a match rule set, add the following parameters to the `MDMBDRMMatchRuleSet` section within the `MDMBDRMMatchRulesSet` section:

Id

Identifier for the match rule set.

Name

Name for the match rule set.

Population

Optional. Name of the population to use for the match process. If you want to include multiple populations for the match process, create additional match rule sets for the populations that you want to include.

Use the population file name without the extension as the parameter value. By default, the MapReduce jobs use the population that you specify in the configuration file.

The following sample shows the match rule set definitions for the Arabic and Chinese populations:

```
<MDMBDRMMatchRulesSet>
   <MDMBDRMMatchRuleSet Id="00" Name="WI04NAR" Population="arabic_r">
      <IndexingConfiguration>
         <indexFieldName>IDS_name,IDS_alias1,IDS_alias2</indexFieldName>
         <indexType>FUZZY</indexType>
         <keyField>Person_Name</keyField>
         <keyLevel>Extended</keyLevel>
         <AdditionalControl>NAMEFORMAT=L</AdditionalControl>
      </IndexingConfiguration>
      <SearchConfiguration>
         <searchType>WI04NARABIC</searchType>
         <searchFieldName>IDS_name,IDS_alias1,IDS_alias2</searchFieldName>
         <keyField>Person_Name</keyField>
         <searchLevel>Exhaustive</searchLevel>
         <AdditionalControl>NAMEFORMAT=L</AdditionalControl>
      </SearchConfiguration>
      <LWMMatchConfiguration>
         <Purpose>Person_Name</Purpose>
         <MatchLevel>Typical</MatchLevel>
         <Threshold>85</Threshold>
         <MatchField>
            <MField name="Person_Name" type="Fuzzy">fullName</MField>
         </MatchField>
      </LWMMatchConfiguration>
      <MatchRuleSet>
         <MatchConfiguration MatchRuleID="WI04NAR" AutoMergeInd="yes">
            <Purpose>Person_Name</Purpose>
            <MatchLevel>Typical</MatchLevel>
            <Threshold>70</Threshold>
            <MatchField>
               <MField name="Person_Name" type="Fuzzy">IDS_name</MField>
               <MField name="Person_Name" type="Fuzzy">IDS_alias1</MField>
            </MatchField>
            <AdditionalControl />
         </MatchConfiguration>
      </MatchRuleSet>
   </MDMBDRMMatchRuleSet>
   <MDMBDRMMatchRuleSet Id="01" Name="WI04NCH" Population="chinese_r">
      <IndexingConfiguration>
         <indexFieldName>IDS_name</indexFieldName>
         <indexType>User</indexType>
         <PARTITION_COLUMN_NAME length="5" part_of_rowkey="YES">Zip</
```

```
PARTITION_COLUMN_NAME>
      </IndexingConfiguration>
      <SearchConfiguration>
         <searchType>WI04NCH</searchType>
         <searchFieldName>IDS_name</searchFieldName>
      </SearchConfiguration>
      <MatchRuleSet>
         <MatchConfiguration MatchRuleID="WI04NCH" AutoMergeInd="yes">
            <MatchField>
               <MField name="Person_Name" type="Exact">IDS_name</MField>
               <MField name="Person_Name" type="Exact">IDS_alias1</MField>
            </MatchField>
         </MatchConfiguration>
      </MatchRuleSet>
   </MDMBDRMMatchRuleSet>
</MDMBDRMMatchRulesSet>
```

Configuring the Indexes

You can define one or more indexes in the matching rules file. You can define an index within the `MDMBDRMMatchRuleSet` section. You must create multiple `MDMBDRMMatchRuleSet` sections to define multiple indexes.

To define an index, add the following parameters to the `IndexingConfiguration` section within the `MDMBDRMMatchRuleSet` section:

indexFieldName

Name of the columns based on which you want to index the records.

Note: Ensure that you specify the column names in the `PZMAP` section of the configuration file.

For example, `<indexFieldName>IDS_name`.

indexType

Type of index that you want to create. Use one of the following values:

- FUZZY. A heavy index that contains fuzzy keys.
- USER. A lightweight index that contains exact values from the field.

keyField

Required only if you use fuzzy keys for indexing. Name of the SSA-NAME3 field based on which you want to build keys.

keyLevel

Required only if you use fuzzy keys for indexing. Type of key level to build. Use one of the following values:

- Standard. Builds more variations than limited key level but uses less disk space than extended key level.
- Extended. Builds more variations than standard key level and uses more disk space than standard and limited key levels.
- Limited. Builds less variations and uses low disk space than standard and extended key levels.

AdditionalControl

Optional. Additional attributes to configure. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.

- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

PARTITION_COLUMN_NAME

Optional. Indicates the name of the column based on which you can create a partition identifier and add the partition identifier to the key. Use the following attributes to define the partition identifier:

- length. Defines the length of the column that you can add as the prefix to the keys. The maximum length of the column that you can use is 8 bytes. Default is 2 bytes.
- part_of_rowkey. Indicates whether the key includes the partition identifier. Set to true if you want to prefix the key with the partition identifier, and set to false if you do not want to prefix the key with the partition identifier.

Note: If you configure the `PARTITION_COLUMN_NAME` parameter for the initial linking job or initial clustering job, you must configure the `PARTITION_COLUMN_NAME` parameter when you run other jobs to update or increment the initial data.

The following sample shows an index definition for the PersonFullName column:

```
<IndexingConfiguration>
   <indexFieldName>PersonFullName</indexFieldName>
   <indexType>FUZZY</indexType>
   <keyField>Person_Name</keyField>
   <keyLevel>Standard</keyLevel>
   <AdditionalControl/>
   <PARTITION_COLUMN_NAME length="4" part_of_rowkey="Yes">ColumnName1</
PARTITION_COLUMN_NAME>
</IndexingConfiguration>
```

Configuring the Searches

You can define one or more searches for each index in the matching rules file. You can define searches within the `MDMBDRMMatchRuleSet` section.

To define a search, add the following parameters to the `SearchConfiguration` section within the `MDMBDRMMatchRuleSet` section:

searchType

Name for the search that you configure.

searchFieldName

Name of the columns based on which you want to generate key ranges to search records. If you specify multiple columns, use commas to separate them.

For example, `<searchFieldName>IDS_name,IDS_alias1,IDS_alias2</searchFieldName>`.

keyField

Required only if you use fuzzy keys for indexing. Name of the SSA-NAME3 field based on which you want to search records.

Note: You can specify only one field as a key field in a configuration file. If you want to use other fields, you must configure another search .

searchLevel

Required only if you use fuzzy keys for indexing. Type of search that you want to perform. Use one of the following values:

- Narrow
- Typical

- Exhaustive
- Extreme

A search returns more candidates as the search level increases but uses more resources and takes a longer execution time.

AdditionalControl

Optional. Additional attributes that you want to configure. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.
- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

You can add additional `SearchConfiguration` sections within the `MDMBDRMMatchRuleSet` section to define multiple searches.

The following sample shows a search definition named Household:

```
<SearchConfiguration>
   <searchType>Household</searchType>
   <searchFieldName>PersonFullName</searchFieldName>
   <keyField>Person_Name</keyField>
   <searchLevel>Typical</searchLevel>
   <AdditionalControl/>
</SearchConfiguration>
```

Configuring the Matching Rules

You can define one or more matching rules for an index in the matching rules file. You can define the matching rules within the `MDMBDRMMatchRuleSet` section.

To define a matching rule, add the following parameters to the `MatchRuleSet` section within the `MDMBDRMMatchRuleSet` section:

MatchConfiguration

Includes a matching rule and its properties. Use multiple `MatchConfiguration` sections within the `MatchRuleSet` section to configure multiple rules.

You can set the following properties for a matching rule:

- MatchRuleID. Unique name for the match rule. The name cannot exceed 14 characters.
- AutoMergeInd. Indicates whether to merge the matching records manually or automatically. Set to Yes for automatic merging, and set to No for manual merging.

Purpose

Required for fuzzy matching. Type of purpose that you want to use for matching. You can use one of the following standard SSA-NAME3 purposes:

- Address. Identifies an address match.
- Contact. Identifies a contact within an organization at a specific location.
- Division. Identifies an organization at an address.
- Fields. Identifies generic data.
- Household. Identifies individuals with same or similar family names who share the same address.
- Individual. Identifies a specific individual by name, ID, or date of birth.
- Organization. Identifies an organization by name.

- Person_Name. Identifies a person by name.
- Resident. Identifies a person at an address.
- Wide_Contact. Identifies a contact within an organization.

MatchLevel

Required for fuzzy matching. Level of matching that you want to perform. Use one of the following values:

- Typical. Returns more results that the conservative match level and less results than the loose match level.
- Conservative. Returns almost accurate results, and you can use in environments where the accuracy of a match is important.
- Loose. Returns matches with more variations than typical match, and you can use in environments where you can manually review the results.

Threshold

Required for fuzzy matching. Minimum match score to consider a record as a matching record.

Lower Threshold

Optional. Minimum match score to consider a record as a probable matching record. If a match score is between the threshold and lower threshold values, the record is considered as a probable matching record.

If you do not specify a lower threshold value, the matching process does not identify any probable matching records.

MatchField

Maps the SSA-NAME3 fields with the input record fields and sets the properties for each field. You can set the following properties for each field:

- name. Indicates the SSA-NAME3 field for the input record field.
- type. Indicates the type of matching to perform on the field. Set to Fuzzy to perform fuzzy matching, and set to Exact to perform exact matching on the field values.
- segment_ind. Optional. Indicates whether you want to enable segment matching. Use the segment matching to match the records that contain any of the specified values for a field. Set to 1 to enable segment matching and 0 to disable segment matching. Default is 0.

  Note: If you enable segment matching for a field, the matching process ignores the null matching or non-equal matching configuration.
- segment_val. Optional. List of values based on which you want to perform segment matching. Specify the `segment_val` parameter only when you enable segment matching.

  The following sample code configures segment matching for the `City` field:

  ```
  <MField name="City" type="Exact" segment_ind="1" segment_val="New
  York,Tokyo">City</MField>
  ```

  The preceding sample code matches the records that have New York or Tokyo as the `City` field values.

  If you want to include null to the list of values, add a comma before the first segment value. For example, `segment_val=",New York,Tokyo"`

- null_ind. Optional. Indicates how the matching process must handle null values. You can set the null_ind property to one of the following values:
  - 0. Does not match a null value with any other values. Default is 0.
  - 1. Considers only two null values as a match and does not match any other values. If you want the match rule to match other values, create another similar rule and set the null_ind property to 0.
  - 2. Considers only a null value and a non-null value as a match and does not match any other values. If you want the match rule to match other values, create another similar rule and set the null_ind property to 0.

  During the linking process, two matched records are linked to the same cluster. When you set null_ind=2 and if one of the linked records contains a null value, the matching process excludes it from further matching.

  For example, consider the following records:
  - 100 John Smith Redwood City
  - 200 John Smith
  - 300 John Smith Las Vegas
  - 400 John Smith Toronto

  When you set null_ind=2, record 100 matches with record 200, and the records then link to cluster 1. The matching process excludes record 200 from further matching. Record 300 and record 400 do not match with record 100, and separate clusters are created for record 300 and record 400.

  Note: If you enable segment matching, the null matching configuration is ignored.
- anti_ind. Optional. Indicates whether you want to perform non-equal matching. Use the non-equal matching to prevent equal values of a field from matching each other and return a successful match only when the values do not match. Set to 1 to enable non-equal matching and 0 to disable non-equal matching. Default is 0.

  Note: If you enable segment matching, the non-equal matching configuration is ignored.

AdditionalControl

Optional. Additional attributes that you want to specify. You can specify the following attributes:

- NAMEFORMAT=L|R. Indicates whether the major word in a name or address is on the left end or the right end. For example, in Western names, the family name is on the right end of the names.
- UNICODE_ENCODING. Specifies the Unicode format of the data that you use.

You can define additional `MatchConfiguration` sections within the `MatchRuleSet` section to define multiple matching rules.

The following sample shows a matching rule definition named Rule1:

```
<MatchRuleSet>
   <MatchConfiguration MatchRuleID="Rule1" AutoMergeInd="yes">
      <Purpose>Fields</Purpose>
      <MatchLevel>Conservative</MatchLevel>
      <Threshold>70</Threshold>
      <LowerThreshold>60</LowerThreshold>
      <MatchField>
         <MField name="ID" type="Exact" segment_ind="1" segment_val="M">PersonSSN</
MField>
         <MField name="Person_Name" type="Exact" null_ind="1"
anti_ind="0">PersonFirstName</MField>
         <MField name="Person_Name" type="Exact" null_ind="1"
anti_ind="0">PersonLastName</MField>
         <MField name="Address_Part1" type="Fuzzy">ShippingAddress</MField>
         <MField name="Telephone_Number" type="Exact">ShippingTelephoneNumber</MField>
      </MatchField>
      <AdditionalControl />
   </MatchConfiguration>
</MatchRuleSet>
```

Configuring the Lightweight Matching Rule

Lightweight matching uses a fast score estimate to reject the obvious unmatched records. SSA-NAME3 performs full matching only on the records that pass the lightweight matching rule, which results in improved matching performance. You can define one lightweight matching rule for an index in the matching rules file within the `MDMBDRMMatchRuleSet` section.

To define a lightweight matching rule, add the following parameters to the `LWMMatchConfiguration` section within the `MDMBDRMMatchRuleSet` section:

Purpose

Type of purpose that you want to use for lightweight matching. You can use one of the following standard SSA-NAME3 purposes:

- Address. Identifies an address match.
- Contact. Identifies a contact within an organization at a specific location.
- Division. Identifies an organization at an address.
- Fields. Identifies generic data.
- Household. Identifies individuals with same or similar family names who share the same address.
- Individual. Identifies a specific individual by name, ID, or date of birth.
- Organization. Identifies an organization by name.
- Person_Name. Identifies a person by name.
- Resident. Identifies a person at an address.
- Wide_Contact. Identifies a contact within an organization.

MatchLevel

Optional. Level of lightweight matching that you want to perform. Use one of the following values:

- Typical. Returns more results that the conservative match level and less results than the loose match level.
- Conservative. Returns almost accurate results, and you can use in environments where the accuracy of a match is important.
- Loose. Returns matches with more variations than typical match, and you can use in environments where you can manually review the results.

Default is Typical.

Threshold

Minimum score required for a record to pass a lightweight matching rule.

MatchField

Maps the SSA-NAME3 fields with the input record fields and sets the properties for each field. You can set the following properties for each field:

- name. Indicates the SSA-NAME3 field for the input record field.
- type. Indicates the type of matching to perform on the field. Set to Fuzzy to perform fuzzy matching, and set to Exact to perform exact matching on the field values.

The following sample configuration uses the `Person_Name` purpose for lightweight matching:

```
<LWMMatchConfiguration>
   <Purpose>Person_Name</Purpose>
   <MatchLevel>Typical</MatchLevel>
   <Threshold>85</Threshold>
   <MatchField>
      <MField name="Person_Name" type="Fuzzy">fullName</MField>
   </MatchField>
</LWMMatchConfiguration>
```

Configuring Additional Parameters

You can configure additional parameters, such as the debug parameter. You can define the additional parameters within the `MDMBDRMEnablerOptions` section.

You can add the following additional parameters to the `MDMBDRMEnablerOptions` section within the `MDMBDRMMatchRulesSet` section:

MaxGroupSize

Maximum number of records a key range can contain. If the number of records exceeds the specified number, the additional records are grouped as poor quality data.

Debug

Indicates whether to perform the MULTISEARCH operation in the debug mode. The debug mode returns performance metrics for the MULTISEARCH operation. Use the debug mode for troubleshooting purposes. Set to true to enable the debug mode, and set to false to disable the debug mode.

The following sample shows the additional parameters that you can configure:

```
<MDMBDRMEnablerOptions>
   <MaxGroupSize>10000</MaxGroupSize>
   <Debug>true</Debug>
</MDMBDRMEnablerOptions>
```

Creating a Consolidation Rules File

You can create a consolidation rules file in the XML format to define the row, column, and default rules. The consolidation process uses the rules to create a preferred record for each cluster.

You can customize the following sample consolidation rules file based on your requirement: `/usr/local/mdmbdrm-<Version Number>/sample/MDMBDRMConsolidationRuleTemplate.xml`

Row Rules

A row rule analyzes the records in a cluster and selects rows of records that match the rule. You can use multiple rules to filter the records to get a single row of preferred record.

If the row rules return more than one record, the consolidation process uses the user-defined default rules to get a single row of preferred record. If you do not define the default rules, the consolidation process uses the system-defined default rules. If the default rules do not identify a single row of record, the consolidation process skips the cluster.

You can configure a name for each rule and set the rule type and execution order. The consolidation process runs the rules based on the execution order until the consolidation process selects a single row of preferred record.

You can use the following row rules:

- RANK
- MODAL_EXACT
- MOST_FILLED
- MOST_DATA
- MAX_COLUMN
- MIN_COLUMN
- EQUALS_COLUMN

The following sample lists some row rules:

```
<RowRules>
   <RowRule name="rowlevel1" rule="MOST_DATA" order="1" />
   <RowRule name="rowlevel3" rule="MOST_FILLED" order="3" />
   <RowRule name="rowLevel2" rule="RANK" order="2">
      <ReferenceColumn columnName="CITY" />
      <Value columnValue="TORONTO" weight="10" />
      <Value columnValue="BANGALORE" weight="90" />
   </RowRule>
   <RowRule name="rowLevel5" rule="MAX_COLUMN" order="5">
      <ReferenceColumn columnValue="AGE" />
   </RowRule>
   <RowRule name="rowLevel4" rule="EQUALS_COLUMN" order="4">
      <ReferenceColumn columnName="AGE">50</ReferenceColumn>
   </RowRule>
   <RowRule name="rowlevel7" rule="MODAL_EXACT" order="7" />
   <RowRule name="rowLevel6" rule="MIN_COLUMN" order="6">
      <ReferenceColumn columnName="AMOUNT" />
   </RowRule>
   <RowRule name="rowLevel8" rule="RANK" order="8">
      <ReferenceColumn columnName="LMT_SOURCE_NAME" />
      <Value columnValue="Salesforce" weight="10" />
      <Value columnValue="ERP" weight="90" />
   </RowRule>
</RowRules>
```

RANK

The RANK rule selects a row based on the specified column value that has the highest weight.

For example, set the weight for the Sales department to 90 and the weight for the Support department to 10.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the RANK rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	New York	31	Sales	Row-level preferred record
1	J Smith		32	Human Resources	
1	John Jr Smith	Seattle		Support	

The RANK rule selects the first row as the preferred record because it contains Sales that has the highest weight.

Use the following format to configure a RANK rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="RANK" order="<Execution order>">
```

```
      <ReferenceColumn columnName="<Column name>" />
      <Value columnValue="<Column value 1>" weight="<Weight 1>" />
      <Value columnValue="<Column value 2>" weight="<Weight 2>" />
      ...
      <Value columnValue="<Column value N>" weight="<Weight N>" />
   </RowRule>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Column name

Name of the column based on which you want to configure the rule.

Column value 1, 2,...N

Column values for which you want to set the weight.

Weight 1, 2,...N

Weight for the corresponding column value. The consolidation process uses the weight of the column value to identify the preferred records.

The following sample RULE rule sets the weight for the Sales department to 90 and the weight for the Support department to 10:

```
<RowRules>
   <RowRule name="rowLevel2" rule="RANK" order="1">
      <ReferenceColumn columnName="Department" />
      <Value columnValue="Support" weight="10" />
      <Value columnValue="Sales" weight="90" />
   </RowRule>
</RowRules>
```

MODAL_EXACT

The MODAL_EXACT rule selects the row that has the highest number of columns with the most frequent non-blank values.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the MODAL_EXACT rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	New York	31	Sales	Row-level preferred record
1	J Smith	New York	31	Human Resources	
1	John Smith	Seattle	30	Sales	

The MODAL_EXACT rule selects the first row as the preferred record because it has four columns with the most frequent non-blank values.

Use the following format to configure a MODAL_EXACT rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="MODAL_EXACT" order="<Execution order>">
   </RowRule>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

The following sample uses the MODAL_EXACT rule:

```
<RowRules>
   <RowRule name="rowlevel7" rule="MODAL_EXACT" order="1" />
</RowRuless>
```

MOST_DATA

The MOST_DATA rule selects the row that has the highest character count.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the MOST_DATA rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Jose	31	Sales	
1	J Smith	Redwood City	30	Human Resources	Row-level preferred record
1	John Jr Smith	Seattle		Support	

The MOST_DATA rule selects the second row as the preferred record because it contains the highest character count compared with other rows.

Use the following format to configure a MOST_DATA rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="MOST_DATA" order="<Execution order>"/>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

The following sample uses the MOST_DATA rule:

```
<RowRules>
   <RowRule name="rowlevel1" rule="MOST_DATA" order="1" />
</RowRules>
```

MOST_FILLED

The MOST_FILLED rule selects the row that has the highest number of non-blank columns.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the MOST_FILLED rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith		31	Sales	
1	J Smith	New York	30	Human Resources	Row-level preferred record
1	John Jr Smith	Seattle		Support	

The MOST_FILLED rule selects the second row as the preferred record because it contains values in four columns.

Use the following format to configure a MOST_FILLED rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="MOST_FILLED" order="<Execution order>"/>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

The following sample includes the MOST_FILLED rule:

```
<RowRules>
   <RowRule name="rowlevel3" rule="MOST_FILLED" order="1" />
</RowRules>
```

MAX_COLUMN

The MAX_COLUMN rule selects the row that contains the highest value in the specified column. The MAX_COLUMN rule uses case-insensitive lexicographical ordering to find the highest value.

The following table displays a sample cluster and indicates the preferred record for the cluster based on age:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	31	Sales	
1	J Smith	Redwood City	30	Human Resources	
1	John Jr Smith	Seattle	32	Support	Row-level preferred record

The MAX_COLUMN rule selects the third row as the preferred record because it contains the highest age.

Use the following format to configure a MAX_COLUMN rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="MAX_COLUMN" order="<Execution order>">
      <ReferenceColumn columnName="<Column name>" />
   </RowRule>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Column name

Name of the column based on which you want to configure the rule.

The following sample MAX_COLUMN rule is configured for the AGE column:

```
<RowRules>
   <RowRule name="rowLevel5" rule="MAX_COLUMN" order="1">
      <ReferenceColumn columnName="AGE" />
   </RowRule>
</RowRules>
```

MIN_COLUMN

The MIN_COLUMN rule selects the row that contains the lowest value in the specified column. The MIN_COLUMN rule uses case-insensitive lexicographical ordering to find the lowest value.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the donation amount:

Cluster ID	Employee Name	City	Donation Amount	Department	Comment
1	John Smith	San Francisco	800	Sales	
1	J Smith	Redwood City	1000	Human Resources	Row-level preferred record
1	John Jr Smith	Seattle	900	Support	

The MIN_COLUMN rule selects the second row as the preferred record because 1000 is the lowest donation amount based on the lexicographical ordering.

Use the following format to configure a MIN_COLUMN rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="MIN_COLUMN" order="<Execution order>">
      <ReferenceColumn columnName="<Column name>" />
   </RowRule>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Column name

Name of the column based on which you want to configure the rule.

The following sample MIN_COLUMN rule is configured for the Donation Amount column:

```
<RowRules>
   <RowRule name="rowLevel5" rule="MIN_COLUMN" order="1">
      <ReferenceColumn columnName="Donation Amount" />
   </RowRule>
</RowRules>
```

EQUALS_COLUMN

The EQUALS_COLUMN rule selects the row that contains the matching value in the specified column.

For example, set the matching value to Sales for the Department column.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the EQUALS_COLUMN rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	30	Sales	Row-level preferred record
1	J Smith	Redwood City	31	Human Resources	
1	John Jr Smith	Seattle	32	Support	

The EQUALS_COLUMN rule selects the first row as the preferred record because the Department column contains Sales.

Use the following format to configure a EQUALS_COLUMN rule:

```
<RowRules>
   <RowRule name="<Rule name>" rule="EQUALS_COLUMN" order="<Execution order>">
      <ReferenceColumn columnName="<Column name>"><Matching value></
ReferenceColumn>
   </RowRule>
</RowRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Column name

Name of the column based on which you want to configure the rule.

Matching value

Value based on which you want to identify the preferred record. The consolidation process identifies the records that match the specified value.

The following sample EQUALS_COLUMN rule identifies the matching records that contain Sales as the Department column value:

```
<RowRules>
   <RowRule name="rowLevel5" rule="EQUALS_COLUMN" order="5">
      <ReferenceColumn columnName="Department">Sales</ReferenceColumn>
   </RowRule>
</RowRules>
```

Column Rules

A column rule analyzes a column in the cluster and selects the column values that match the rule. If the rule returns more than one column value, the consolidation process uses the user-defined default rules to get a single preferred column value. If you do not define the default rules, the consolidation process uses the system-defined default rules. If the default rules return more than one preferred column value, the consolidation process skips the cluster.

The preferred column value replaces the corresponding column value in the preferred record that the row rules return. You can configure a column rule for a single column or a group of columns with a name and the rule type.

You can use the following column rules:

- RANK
- MOST_DATA
- MODAL_EXACT
- MAX_COLUMN
- MIN_COLUMN
- FROM_MASTER
- EQUALS_OTHER_COLUMN
- MIN_OTHER_COLUMN
- MAX_OTHER_COLUMN

The following sample uses the column rules:

```
<ColumnRules>
   <ColumnGroup name="COL_GROUP1" rule="MOST_DATA">
      <ColumnRule columnName="NAME" />
      <ColumnRule columnName="CITY" />
   </ColumnGroup>
   <ColumnRule name="OTHERMAX_RULE1" rule="MAX_OTHER_COLUMN" columnName="OTHERCOLM">
      <ReferenceColumn columnName="AMOUNT" />
   </ColumnRule>
   <ColumnRule name="OTHERMIN_RULE1" rule="MIN_OTHER_COLUMN" columnName="OTHERCOLMIN">
      <ReferenceColumn columnName="AGE" />
   </ColumnRule>
   <ColumnRule name="OTHEREQ_RULE1" rule="EQUALS_OTHER_COLUMN" columnName="OTHERCOLEQ">
      <ReferenceColumn columnName="ADDRESS">BANGALORE</ReferenceColumn>
   </ColumnRule>
   <ColumnRule name="AMOUNT_RULE1" rule="MIN_COLUMN" columnName="AMOUNT" />
   <ColumnRule name="SOURCE_RULE" rule="FROM_MASTER" columnName="LMT_SOURCE_NAME" />
   <ColumnRule name="CITY_RULE" rule="RANK" columnName="CITY">
      <Value columnValue="TORONTO" weight="10" />
      <Value columnValue="BANGALORE" weight="90" />
   </ColumnRule>
   <ColumnRule name="AGE_RULE" rule="MOST_DATA" columnName="AGE" />
   <ColumnRule name="AGE_RULE1" rule="MAX_COLUMN" columnName="AGE" />
   <ColumnRule name="NAME_RULE" rule="MODAL_EXACT" columnName="NAME" />
</ColumnRules>
```

RANK

The RANK rule selects a column value based on the specified column value that has the highest weight.

For example, set the weight for the Human Resources department to 90 and the weight for the Sales department to 10.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	New York	31	Sales	Row-level preferred record
1	J Smith		30	Human Resources	
1	John Jr Smith	Seattle		Support	

The following table lists the preferred record with the preferred column value for the cluster based on the RANK rule:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	New York	31	Human Resources	Preferred record

Human Resources is the preferred column value because it has the highest weight.

Use the following format to configure a RANK rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="RANK" columnName="<Column name">
      <Value columnValue="<Column value 1>" weight="<Weight 1>" />
      <Value columnValue="<Column value 2>" weight="<Weight 2>" />
      ...
      <Value columnValue="<Column value N>" weight="<Weight N>" />
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

Column values 1, 2,...N

Column values for which you want to set the weight.

Weight 1, 2,...N

Weight for the corresponding column value. The consolidation process identifies the preferred column value based on the weight of the column values.

The following sample RULE rule sets the weight for the Human Resources department to 90 and the weight for the Sales department to 10:

```
<ColumnRules>
   <ColumnRule name="DepartmentRule" rule="RANK" columnName="Department">
      <Value columnValue="Human Resources" weight="90" />
      <Value columnValue="Sales" weight="10" />
   </ColumnRule>
</ColumnRules>
```

MOST_DATA

The MOST_DATA rule selects the column value that has the highest character count.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Jose	31	Sales	
1	J Smith	Redwood City	30	Human Resources	Row-level preferred record
1	John Jr Smith	Seattle		Support	

The following table lists the preferred record with the preferred column value for the cluster based on the MOST_DATA rule applied on the Employee Name column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Jr Smith	Redwood City	30	Human Resources	Column-level preferred record

John Jr Smith is the preferred column value because it has the highest character count.

Use the following format to configure a MOST_DATA rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MOST_DATA" columnName="<Column name">
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

The following sample MOST_DATA rule is based on the Employee Name column:

```
<ColumnRules>
   <ColumnRule name="NAME_RULE" rule="MOST_DATA" columnName="Employee Name"></ColumnRule>
</ColumnRules>
```

MODAL_EXACT

The MODAL_EXACT rule selects a column value that has the highest count of the most frequent non-blank values.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	New York	31	Sales	Row-level preferred record
1	J Smith	Seattle	31	Human Resources	
1	John Smith	Seattle	30	Sales	

The following table lists the preferred record with the preferred column value based on the MODAL_EXACT rule applied on the City column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	Seattle	31	Sales	Column-level preferred record

Seattle is the preferred column value because it has the highest count of the most frequent value.

Use the following format to configure a MODAL_EXACT rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MODAL_EXACT" columnName="<Column name">
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

The following sample MODAL_EXACT rule is based on the City column:

```
<ColumnRules>
   <ColumnRule name="NAME_RULE" rule="MODAL_EXACT" columnName="City"></ColumnRule>
</ColumnRules>
```

MAX_COLUMN

The MAX_COLUMN rule selects the highest column value. The MAX_COLUMN rule uses case-insensitive lexicographical ordering to find the highest value.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Donation Amount	Department	Comment
1	John Smith	San Francisco	1000	Sales	
1	J Smith	Redwood City	800	Human Resources	
1	John Jr Smith	Seattle	900	Support	Row-level preferred record

The following table lists the preferred record with the preferred column value based on the MAX_COLUMN rule applied on the Donation Amount column:

Cluster ID	Employee Name	City	Donation Amount	Department	Comment
1	John Jr Smith	Seattle	900	Support	Column-level preferred record

The MAX_COLUMN rule selects 900 as the preferred column value because it is the highest donation amount based on the lexicographical ordering.

Use the following format to configure a MAX_COLUMN rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MAX_COLUMN" columnName="<Column name">
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

The following sample MAX_COLUMN rule is based on the Donation Amount column:

```
<ColumnRules>
   <ColumnRule name="NAME_RULE" rule="MAX_COLUMN" columnName="Donation Amount"></
ColumnRule>
</ColumnRules>
```

MIN_COLUMN

The MIN_COLUMN rule selects the lowest column value. The MIN_COLUMN rule uses case-insensitive lexicographical ordering to find the lowest value.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	31	Sales	
1	J Smith	Redwood City	30	Human Resources	
1	John Jr Smith	Seattle	32	Support	Row-level preferred record

The following table lists the preferred record with the preferred column value based on the MIN_COLUMN rule applied on the Age column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Jr Smith	Seattle	30	Support	Column-level preferred record

The MIN_COLUMN rule selects 30 as the preferred column value because it is the lowest age.

Use the following format to configure a MIN_COLUMN rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MIN_COLUMN" columnName="<Column name">
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

The following sample MIN_COLUMN rule is based on the Age column:

```
<ColumnRules>
   <Column name="NAME_RULE" rule="MIN_COLUMN" columnName="Age"></ColumnRule>
</ColumnRules>
```

FROM_MASTER

The FROM_MASTER rule retains the column value in the preferred record that the row rules have selected.

The following table displays a sample cluster and indicates the row-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	31	Sales	
1	J Smith	Redwood City	30	Human Resources	Row-level preferred record
1	John Jr Smith	Seattle	32	Support	

The following table lists the preferred record with the preferred column value based on the FROM_MASTER rule applied on the Age column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	J Smith	Redwood City	30	Human Resources	Column-level preferred record

The FROM_MASTER rule selects 30 as the preferred column value.

Use the following format to configure a FROM_MASTER rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="FROM_MASTER" columnName="<Column name">
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Column name

Name of the column based on which you want to configure the rule.

The following sample FROM_MASTER rule is based on the Age column:

```
<ColumnRules>
   <ColumnRule name="NAME_RULE" rule="FROM_MASTER" columnName="Age"></ColumnRule>
</ColumnRules>
```

EQUALS_OTHER_COLUMN

The EQUALS_OTHER_COLUMN rule selects a column value from a row that fulfills the specified condition. The specified condition can be based on another column.

For example, use the City value of a record as the preferred column value if the Department column value of the record is equal to Human Resources.

The following table displays a sample cluster and indicates the rule-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	31	Sales	Row-level preferred record
1	J Smith	Redwood City	32	Human Resources	
1	John Jr Smith	Seattle	30	Support	

The following table lists the preferred record with the preferred column value based on the EQUALS_OTHER_COLUMN rule applied on the City column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	Redwood City	31	Sales	Column-level preferred record

Redwood City is the preferred column value because it is part of the row that contains Human Resources as the Department column value.

Use the following format to configure an EQUALS_OTHER_COLUMN rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="EQUALS_OTHER_COLUMN" columnName="<Primary column
name">
      <ReferenceColumn columnName="Reference column name"><Matching value></
ReferenceColumn>
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Primary column name

Name of the column based on which you want to configure the rule.

Reference column name

Name of the column in which you want to search for the matching value.

Matching value

Value to search in the secondary column. The consolidation process identifies the record that contains the specified column value and then uses the primary column value from the selected record as the preferred column value.

The following sample EQUALS_OTHER_COLUMN rule indicates to use the City value as the preferred column value if the Department column value is equal to Human Resources:

```
<ColumnRules>
   <ColumnRule name="OTHEREQ_RULE1" rule="EQUALS_OTHER_COLUMN" columnName="City">
      <ReferenceColumn columnName="Department">Human Resources</ReferenceColumn>
   </ColumnRule>
</ColumnRules>
```

MAX_OTHER_COLUMN

The MAX_OTHER_COLUMN rule selects a column value from a record that contains the highest value of the specified column. The MAX_OTHER_COLUMN rule uses case-insensitive lexicographical ordering to find the highest value.

For example, use the City value from a record that contains the highest donation amount.

The following table displays a sample cluster and indicates the rule-level preferred record:

Cluster ID	Employee Name	City	Donation Amount	Department	Comment
1	John Smith	San Francisco	1000	Sales	Row-level preferred record
1	J Smith	Redwood City	1900	Human Resources	
1	John Jr Smith	Seattle	800	Support	

The following table lists the preferred record with the preferred column value based on the MAX_OTHER_COLUMN rule applied on the City column:

Cluster ID	Employee Name	City	Donation Amount	Department	Comment
1	John Smith	Redwood City	1000	Sales	Column-level preferred record

Redwood City is the preferred column value because it is part of the row that contains the highest donation amount.

Use the following format to configure a MAX_OTHER_COLUMN rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MAX_OTHER_COLUMN" columnName="<Primary column
name">
      <ReferenceColumn columnName="<Reference column name>" />
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Primary column name

Name of the column based on which you want to configure the rule.

Reference column name

Name of the column in which you want to find the highest value. The consolidation process identifies the record that contains the highest column value and then uses the primary column value from the selected record as the preferred column value.

The following sample MAX_OTHER_COLUMN rule indicates to use the City value from a record that contains the highest donation amount:

```
<ColumnRules>
   <ColumnRule name="OTHEREQ_RULE1" rule="MAX_OTHER_COLUMN" columnName="City">
      <ReferenceColumn columnName="Donation Amount" />
   </ColumnRule>
</ColumnRules>
```

MIN_OTHER_COLUMN

The MIN_OTHER_COLUMN rule selects the column value from a record that contains the lowest value of the specified column. The MIN_OTHER_COLUMN rule uses case-insensitive lexicographical ordering to find the lowest value.

For example, use the City value from a record that contains the lowest age.

The following table displays a sample cluster and indicates the rule-level preferred record:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	San Francisco	31	Sales	Row-level preferred record
1	J Smith	Redwood City	32	Human Resources	
1	John Jr Smith	Seattle	30	Support	

The following table lists the preferred record with the preferred column value based on the MIN_OTHER_COLUMN rule applied on the City column:

Cluster ID	Employee Name	City	Age	Department	Comment
1	John Smith	Seattle	31	Sales	Column-level preferred record

Seattle is the preferred column value because it is part of the row that contains the lowest age.

Use the following format to configure a MIN_OTHER_COLUMN rule:

```
<ColumnRules>
   <ColumnRule name="<Rule name>" rule="MIN_OTHER_COLUMN" columnName="<Primary column
name">
      <ReferenceColumn columnName="<Reference column name>" />
   </ColumnRule>
</ColumnRules>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Primary column name

Name of the column based on which you want to configure the rule.

Reference column name

Name of the column in which you want to find the lowest value. The consolidation process identifies the record that contains the lowest column value and then uses the primary column value from the selected record as the preferred column value.

The following sample MIN_OTHER_COLUMN rule indicates to use the City value from a record that contains the lowest age:

```
<ColumnRules>
   <ColumnRule name="OTHEREQ_RULE1" rule="MIN_OTHER_COLUMN" columnName="City">
      <ReferenceColumn columnName="Age" />
   </ColumnRule>
</ColumnRules>
```

Configuring a Column Rule for a Group of Columns

You can configure a column rule for a group of columns. The consolidation process identifies the preferred column value for each column based on the rule and updates the corresponding column value in the preferred record that the row rules return.

Use the following format to configure a column rule for a group of columns:

```
<ColumnRules>
   <ColumnGroup name="<Unique name>" rule="<Column rule name>">
      <ColumnRule columnName="<Column name 1>" />
      <ColumnRule columnName="<Column name 2>" />
      ...
      <ColumnRule columnName="<Column name N>" />
   </ColumnGroup>
</ColumnRules>
```

The format uses the following parameters:

Unique name

Unique name for the rule.

Column rule name

Name of the column rule that you want to use.

You can use the following column rules:

- RANK
- MOST_DATA
- MODAL_EXACT
- MAX_COLUMN
- MIN_COLUMN
- FROM_MASTER
- EQUALS_OTHER_COLUMN
- MIN_OTHER_COLUMN
- MAX_OTHER_COLUMN

Column name 1, 2,...N

Name of the columns for which you want to configure the rule.

The following sample MAX_COLUMN rule is configured for the NAME and CITY columns:

```
<ColumnRules>
   <ColumnGroup name="COL_GROUP1" rule="RANK">
      <ColumnRule columnName="NAME" />
      <ColumnRule columnName="CITY" />
   </ColumnGroup>
</ColumnRules>
```

User-Defined Default Rules

You can define default rules to identify a single preferred record when the row rules return multiple records or to identify a single preferred column value when a column rule returns multiple values. If you do not define default rules, the consolidation process uses the system-defined default rules.

You can configure a name for each user-defined rule and set the rule type and execution order. The consolidation process runs the rules based on the execution order until the consolidation process selects a single preferred record or column value.

You can use the following user-defined default rules:

- LATEST_TIMESTAMP
- DEFAULT_RULE_MAX_COLUMN_VALUES

The following sample lists the LATEST_TIMESTAMP and DEFAULT_RULE_MAX_COLUMN_VALUES rules:

```
<DefaultRules>
   <DefaultRule name="defaultRule1" rule="LATEST_TIMESTAMP" order="1">
      <ReferenceColumn columnName="Date&Time" />
   </DefaultRule>
   <DefaultRule name="defaultRule2" rule="DEFAULT_RULE_MAX_COLUMN_VALUES" order="2">
      <Value columnValue="ID" weight="2" />
      <Value columnValue="SAP" weight="1" />
   </DefaultRule>
</DefaultRules>
```

LATEST_TIMESTAMP

The LATEST_TIMESTAMP rule selects the row that contains the most recent value in a time stamp column as the preferred record. To identify a preferred column value, the LATEST_TIMESTAMP rule selects the corresponding column value in the preferred record as the preferred column value.

The time stamp column must contain values in the `yyyyMMddHHmmss` format. If you do not specify a time stamp column for the rule, the consolidation process selects the row with the most recent time stamp in the repository or skips the rule if you persist the consolidated data in HDFS.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the LATEST_TIMESTAMP rule:

Cluster ID	Employee Name	City	Date&Time	Department	Comment
1	John Smith	New York	20140923114923	Sales	Row-level preferred record
1	J Smith		20131020094712	Human Resources	
1	John Jr Smith	Seattle	20120130123446	Support	

The LATEST_TIMESTAMP rule selects the first row as the preferred record because it contains the most recent time stamp value.

Use the following format to configure a LATEST_TIMESTAMP rule:

```
<DefaultRule name="<Rule name>" rule="LATEST_TIMESTAMP" order="<Execution order>">
   <ReferenceColumn columnName="<Time stamp column name>" />
</DefaultRule>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Time stamp column name

Optional. Name of the time stamp column that contains values in the `yyyyMMddHHmmss` format. By default, the consolidation process uses the time stamp of the record in the repository.

The following sample LATEST_TIMESTAMP rule uses the Date&Time column to identify the preferred record:

```
<DefaultRule name="defaultRule1" rule="LATEST_TIMESTAMP" order="1">
   <ReferenceColumn columnName="Date&Time" />
</DefaultRule>
```

DEFAULT_RULE_MAX_COLUMN_VALUES

The DEFAULT_RULE_MAX_COLUMN_VALUES rule selects the row that contains the highest value in the specified columns as the preferred record. To identify a preferred column value, the DEFAULT_RULE_MAX_COLUMN_VALUES rule selects the corresponding column value in the preferred record as the preferred column value.

You must configure weight for each column that you specify. The consolidation process uses the descending order of the column weights to determine the order of the execution of the columns. The DEFAULT_RULE_MAX_COLUMN_VALUES rule uses case-insensitive lexicographical ordering to find the highest value.

For example, set the weight for the EmployeeID column to 5 and weight for the City column to 2.

The following table displays a sample cluster and indicates the preferred record for the cluster based on the DEFAULT_RULE_MAX_COLUMN_VALUES rule:

Cluster ID	Employee Name	City	EmployeeID	Source	Comment
1	John Smith	San Francisco	502342	Oracle	
1	J Smith	Redwood City	702343	SAP	Row-level preferred record
1	John Jr Smith	Seattle	234234	SAP	

The DEFAULT_RULE_MAX_COLUMN_VALUES rule uses the EmployeeID column first because the EmployeeID column has the highest weight. The rule selects the second row as the preferred record because it contains the highest employee ID.

Use the following format to configure a DEFAULT_RULE_MAX_COLUMN_VALUES rule:

```
<DefaultRule name="<Rule name>" rule="DEFAULT_RULE_MAX_COLUMN_VALUES" order="<Execution
order>">
   <Value columnValue="<Column name 1>" weight="<Weight 1>" />
   <Value columnValue="<Column name 2>" weight="<Weight 2>" />
   ...
   <Value columnValue="<Column name N>" weight="<Weight N>" />
</DefaultRule>
```

The format uses the following parameters:

Rule name

Unique name for the rule.

Execution order

Optional. Order of execution for the rule. Use an integer value for the order of execution. If you do not specify the execution order, the consolidation process runs the rule based on the sequence of rules listed in the consolidation rules file. If you specify the execution order for a rule, you must specify the execution order for other rules.

Column name 1, 2,...N

Name of the columns based on which you want to configure the rule.

Weight 1, 2,...N

Weight for the corresponding column. The descending order of the column weights determines the order of execution of the columns.

The following sample DEFAULT_RULE_MAX_COLUMN_VALUES rule is configured for the EmployeeID and City columns:

```
<DefaultRule name="defaultRule2" rule="DEFAULT_RULE_MAX_COLUMN_VALUES" order="1">
   <Value columnValue="EmployeeID" weight="5" />
   <Value columnValue="City" weight="2" />
</DefaultRule>
```

System-Defined Default Rules

The consolidation process uses the system-defined default rules only when you do not define any default rule in the consolidation rules file. The system-defined default rules identify a single preferred record when the row rules return multiple records or a single preferred column value when a column rule returns multiple values.

The system-defined default rules uses the following rules:

1. Recent time stamp of the record. The consolidation process skips the rule when you persist the consolidated data in HDFS or when you run the consolidation job in the initial mode.
2. Highest value of the combined primary key and source column values. The consolidation process uses case-insensitive lexicographical ordering to find the highest value.

For example, if a column rule returns multiple values and you do not define any default rules, the consolidation process uses the system-defined default rules. The rules identify the record that has the recent time stamp and then select the corresponding column value as the preferred column value. If you have multiple records with the same time stamp, the rules combine the primary key and source column values of each record and select the record that has the highest value. The rules then return the corresponding column value as the preferred column value.

Creating a Relationship Configuration File

You can create a relationship configuration file in the XML format to define the business entity types, their potential relationships, and the columns that you want to display in the relationship graph.

You can customize the following sample relationship configuration file based on your requirement: `/usr/local/mdmbdrm-<Version Number>/sample/MDMBDRMRelationshipConfigTemplate.xml`

Configuring the Relationship Metadata

You can configure the metadata information for the relationship graph that you configure for the business entity types. The metadata information includes relationship name, repository host name, and the port number on which the server listens. You can use the same repository that stores the processed data or use a different repository for the relationship graph.

To configure the relationship metadata information, add the following parameters to the `RelationshipMetaData` section in the relationship configuration file:

RelationshipName

Name of the relationship graph that you configure.

HbaseMaster

Host name of the HBase Master server and the port number on which the Master server listens.

Specify the port number based on the `hbase.master.port` parameter configured in HBase.

Use the following format to specify a value for the `HbaseMaster` parameter:

```
<Master Server Host Name>:<Port>
```

HbaseZookeeperQuorum

Comma-separated list of ZooKeeper servers when HBase uses an ensemble of ZooKeeper servers.

For example: server1.domain.com,server3.domain.com

You can get the list of servers from the `hbase.zookeeper.quorum` property in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseZookeeperClientPort

Optional. Port number on which the ZooKeeper server listens for client connections. Default is 2181.

HbaseRootDirectory

Required if you use Hortonworks Data Platform. Directory that the region servers share on the local file system. Default is `${hbase.tmp.dir}/local`.

Note: The `HbaseRootDirectory` parameter overrides the `hbase.rootdir` parameter configured in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

HbaseZookeeperZnodeParent

Required if you use Hortonworks Data Platform. Root ZNode path for the ZooKeeper files. HBase stores all the ZooKeeper files configured with the relative path in the root ZNode path. Default is `/hbase`.

Note: The `HbaseZookeeperZnodeParent` parameter overrides the `zookeeper.znode.parent` parameter configured in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

DriverName

Name of the HBase driver to use. Use `com.informatica.mdmbde.database.hbase.HbaseDatabaseAdapterImplVOne` as the driver name.

The following sample code shows the metadata details for a relationship graph named Customer360:

```
<RelationshipMetaData>
   <RelationshipName>Customer360</RelationshipName>
   <HbaseMaster>localhost:60000</HbaseMaster>
   <HbaseZookeeperClientPort>2181</HbaseZookeeperClientPort>
   <HbaseZookeeperQuorum>localhost</HbaseZookeeperQuorum>
   <DriverName>com.informatica.mdmbde.database.hbase.HbaseDatabaseAdapterImplVOne</
DriverName>
   <HbaseZookeeperZnodeParent />
   <HbaseRootDirectory />
</RelationshipMetaData>
```

Configuring the Relationship Graph

Configure the business entity types that you want to display in the relationship graph. For each business entity type, you can configure the potential relationships, the list of columns based on which you want to display the relationship graph, the URL to access the RESTful web services, and other parameters.

To configure the relationship graph, add the following parameters to the `Relationship` section in the relationship configuration file:

Relationship

Defines the identifiers for the relationship table name. The following batch jobs use the identifiers to form the relationship table name:

- Load match pairs
- Create relationship graph

Use the following attributes:

- name. Unique identifier for the relationship table.
- OrganizationID. Unique identifier for the organization.
- ColumnFamilyName. Identifier for the column family that groups all the columns in the relationship table.

A relationship table name uses the following format:

```
MDMBDRM<OrganizationID>_<name>
```

The following sample code defines a relationship table named `MDMBDRMSAP_RelTable`:

```
<Relationship name="RelTable" OrganizationID="SAP" ColumnFamilyName="cf">
```

BDRMEntities

Lists the business entity types that you plan to use.

BDRMEntity

Properties for a business entity type. The properties include a name and an icon for the business entity type.

Use the following attributes:

- name. Name of the business entity type that you configure.
- classType. Icon for the business entity type. Default is none.

  Specify one of the following values:

Value	Icon
person	👤
house	🏠
cart	🛒
request	📢
none	

The following sample code configures properties for the `Customer` business entity type:

```
<BDRMEntity name="Customer" classType="person">
```

MetaData

Maps the primary key and source columns of the business entity type to the equivalent columns in the other business entity types.

Use the following format to specify the column names:

```
<MetaData>
   <ROWID name="<Column1>"><Column2></ROWID>
   <Source name="<Column3>"><Column4></Source>
</MetaData>
```

`Column1` is mapped to `Column2`, and `Column3` is mapped to `Column4`.

For example, if the primary key column in the Customer business entity type is `ID` and the product business entity type contains the customer identifier in the `customerID` column, map the columns.

The following sample code maps the `ID` and `LMT_SOURCE_NAME` columns to the `customerID` and `customerLMT_SOURCE_NAME` columns:

```
<MetaData>
   <ROWID name="ID">customerID</ROWID>
   <Source name="LMT_SOURCE_NAME">customerLMT_SOURCE_NAME</Source>
</MetaData>
```

ConfigurationXML

Absolute path and file name of the configuration file that you use to process the input records of the business entity type.

RESTEndPoint

URL that accesses the RESTful web services of the business entity type. If you do not specify the URL, you cannot search records in the search page of the relationship graph user interface.

The following URL accesses the RESTful web services of the Customer business entity type:

```
http://localhost:8080/MDMBDRMCustomer360/v4.0/Customer360/
```

Note: The URL is valid only when you end the URL with the slash (/) character.

DisplayField

Name of the column whose values you want to use as node labels in the relationship graph.

EntityViewFields

List of the columns that you want to display in the right pane of the relationship graph when you select a node. You can use the same column names or use alias names for the columns. If you use alias names, the right pane of the relationship graph displays the alias names.

Use the following format to specify the list of columns:

```
<EntityViewFields>
   <Field name="<Field1>"><Field1 Alias></Field>
   <Field name="<Field2>"><Field2 Alias></Field>
   <Field name="<Field3>"><Field3 Alias></Field>
</EntityViewFields>
```

The following sample code uses alias names for the `id` and `Name` columns:

```
<EntityViewFields>
   <Field name="id">Customer ID</Field>
   <Field name="LMT_SOURCE_NAME" />
   <Field name="Name">Full Name</Field>
</EntityViewFields>
```

RelationshipType

List of relationships that the business entity type can use. You can connect a business entity type with another business entity type through a relationship. The relationship types include inbound and outbound relationships.

The following sample code lists the inbound and outbound relationships for a business entity type:

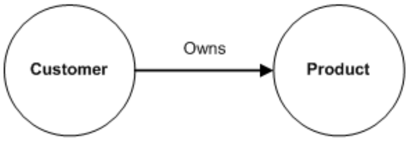
```
<RelationshipType>
   <InBoundRelationship>
      <Label name="Duplicate" />
      <Label name="Household" />
   </InBoundRelationship>
   <OutBoundRelationship>
      <Label name="Duplicate" />
      <Label name="Household" />
   </OutBoundRelationship>
</RelationshipType>
```

OutBoundRelationship

List of the outbound relationships that the business entity type can use. The relationships through which an entity connects to other entities are called outbound relationships.

For example, consider Customer and Product as business entity types. The relationship `Owns` connects the customers and the products they own. In this case, `Owns` is an outbound relationship for the Customer business entity type.

The following image shows the outbound relationship of the Customer business entity type:

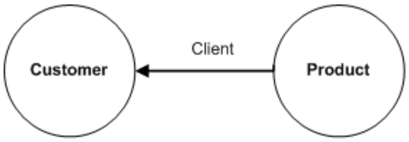


InBoundRelationship

List of the inbound relationships that the business entity type can use. The relationships through which other entities connect to an entity are called inbound relationships.

For example, consider Customer and Product as business entity types. the relationship `Client` connects the products and the customers who own the products. In this case, `Client` is an inbound relationship for the Customer business entity type.

The following image shows the inbound relationship of the Customer business entity type:



Label

Name of the relationship for the business entity type.

EntityFields

List of the columns that you want to use as filters in the relationship graph.

Field

Name of the column and its properties.

Use the following attributes:

- name. Name of the column.
- length. Length of the column. Ensure that the value matches with the length of the column that you specify in the `PZMAP` section of the configuration file.
- filterable. Indicates whether the column can be used to filter the records. Set to `true` to use the column as a filter, and set to `false` if you do not want to use the column as a filter.
- aggregatable. Indicates whether you can aggregate the records that have relationship. When you aggregate the records, the relationship graph shows the total number of related records.

The following sample code configures the properties for the `Age` column:

```
<Field name="Age" length="5" filterable="true" aggregatable="false">
```

Facets

Type of filter for the column. Applicable only when you configure the column as a filter.

You can use the following filter types:

- Exact. Filters the records based on the value that you provide in the relationship graph.
- Enum. Filters the records based on the enumeration values that you configure.
- Range. Filters the records based on the range that you configure.

Use the following format to configure the type of filter:

```
<Facets type="<Filter Type>">
```

For example, the following sample code configures the filter type as `Enum`:

```
<Facets type="Enum">
```

Facet

Enumeration value or range that you want to configure for the column. Applicable only when the filter type is `Enum` or `Range`.

Use the following format to configure enumeration values:

```
<Facet id="<Identifier>"><Value></Facet>
```

For example, the following sample code configures the enumeration values for the `Gender` column:

```
<Field name="Gender" length="2" filterable="true" aggregatable="true">
   <Facets type="Enum">
      <Facet id="M">Male</Facet>
      <Facet id="F">Female</Facet>
   </Facets>
</Field>
```

Use the following format to configure range values:

```
<Facet id="<Identifier>" min="<Minimum Range Value>" max="<Maximum Range Value>"/>
```

For example, the following sample code configures the range for the `Age` column:

```
<Field name="Age" length="5" filterable="true" datatype="integer"
aggregatable="false">
   <Facets type="Range">
      <Facet id="Infant" min="0" max="4" />
      <Facet id="Child" min="5" max="12" />
      <Facet id="Adult" min="13" max="140" />
   </Facets>
</Field>
```

RELATED TOPICS:

- “Deploying the RESTful Web Services” on page 82

Creating a Properties File for the Relationship Graph User Interface

If you want to use the relationship graph user interface, you must create a properties file with parameters related to the relationship graph.

The following sample shows the properties configured for the relationship graph user interface:

```
serverURL=10.19.20.43:8080/MDMBDRMCustomer360/v5.0/Customer360/
defaultPagination=20
graph_depth=3
graph_maxnode=50
```

You can configure the following parameters in the properties file:

serverURL

URL to access the relationship graph user interface WAR file.

Use the following format for the URL:

```
<Host IP Address>:<Port>/<WAR File Name>/<Version Number>/<Configuration File
Identifier>/
```

The URL uses the following parameters:

- `Host IP Address`. IP address of the host machine on which you plan to deploy the relationship graph user interface WAR file.
- `Port`. Port number on which the host listens.
- `WAR File Name`. Name of the WAR file that you plan to generate for the RESTful web services.
- `Version Number`. Version number of the RESTful web services.
- `Configuration File Identifier`. Unique identifier for the configuration file and the matching rules file that you specify when you package the RESTful web services.

For example, `10.19.20.43:8080/MDMBDRMCustomer360/v5.0/Customer360/`

defaultPagination

Optional. Maximum number of search results to return in the user interface when you search for a record. Default is 10.

graph_depth

Optional. Maximum number of child levels to display in the relationship graph. Default is 5.

graph_maxnode

Optional. Maximum number of nodes that you want to display in the relationship graph. Default is 500.

CHAPTER 3

Configuring Security

This chapter includes the following topics:

Security Overview

You can secure data from unauthorized access to protect information privacy and data integrity. You can configure security for the batch jobs and the RESTful web services.

If you have an existing Kerberos authentication infrastructure, you can use the Kerberos authentication for the batch jobs to access data in HBase and Hive. You can use the OpenAM-based authentication system to authenticate the web services.

Configuring Relate 360 to Use Kerberos Authentication

If you use Kerberos for authentication in your environment, you can use the Kerberos authentication for the batch jobs to access data in HBase and Hive. The authentication process prevents any unauthorized access to the data.

Note: Ensure that you have an existing Kerberos authentication infrastructure.

1. Open the configuration file in a text editor.
2. To configure the Kerberos authentication for HBase, ensure that you add the following parameters within the `HBASEConfiguration` section:

   KeyTabFile

   Optional. Absolute path and file name of the keytab file. A keytab file contains a list of keys that are analogous to user passwords. Applicable if you use Kerberos for authentication.

   Note: If you use the `KeyTabFile` parameter, ensure that the name of the file and the absolute path to the file are the same for all the nodes in a distributed Hadoop cluster.

PrincipalName

Required if you use Kerberos for authentication. Service Principal Name (SPN) of the HBase master server. For example, hbase/_Host@realm.com.

You can get the SPN of the HBase master server from the `hbase.master.kerberos.principal` property in the following file: `${HBASE_HOME}/conf/hbase-site.xml`

For example, the following sample code contains the parameters related to the Kerberos authentication for HBase:

```
<HBASEConfiguration>
   <HbaseMaster>HadoopServer:60000</HbaseMaster>
   <HbaseZookeeperClientPort>2181</HbaseZookeeperClientPort>
   <HbaseZookeeperQuorum>iir-hadoop-test1</HbaseZookeeperQuorum>
   <HbaseRootDirectory />
   <HbaseDistributed>true</HbaseDistributed>
   <HbaseZookeeperZnodeParent />
   <HbaseCompressionAlgorithm>SNAPPY</HbaseCompressionAlgorithm>
   <HbaseDataBlockEncoding>PREFIX</HbaseDataBlockEncoding>
   <ScanCacheSize>100000</ScanCacheSize>
   <CacheBlock>false</CacheBlock>
   <AutoFlush>false</AutoFlush>
   <WALonPUT>false</WALonPUT>
   <ScanBatchSize>100</ScanBatchSize>
   <EnableSmallScan>false</EnableSmallScan>
   <RegionSplitSize>8</RegionSplitSize>
   <DriverName>com.informatica.mdmbde.database.hbase.HBaseDatabaseAdapterImpl</
DriverName>
   <SearchTokenValidity>1000</SearchTokenValidity>
   <KeyTabFile>/etc/security/keytabs/hbase.keytab</KeyTabFile>
   <PrincipalName>hbase/_Host@realm.com</PrincipalName>
</HBASEConfiguration>
```

3. To configure the Kerberos authentication for Hive, ensure that you add the following parameters within the `HiveConfiguration` section:

KeyTabFile

Optional. Absolute path and file name of the keytab file. A keytab file contains a list of keys that are analogous to user passwords. Applicable if you use Kerberos for authentication.

Note: If you use the `KeyTabFile` parameter, ensure that the name of the file and the absolute path to the file are the same for all the nodes in a distributed Hadoop cluster.

PrincipalName

Required if you use Kerberos for authentication. Service Principal Name (SPN) of the Hive master server. For example, hive/_Host@realm.com.

You can get the SPN of the Hive master server from the `hive.metastore.kerberos.principal` property in the following file: `${HIVE_HOME}/conf/hive-site.xml`

JDBCUrl

JDBC connection URL to access metadata from Hive.

Use the following format for the JDBC connection URL:

```
jdbc:hive2://<Host Name>:<Port>/default
```

`Host Name` indicates the name of the machine that hosts the Hive master server, and `Port` indicates the port number on which the Hive master server listens.

For example, the following sample code contains the parameters related to the Kerberos authentication for Hive:

```
<HiveConfiguration>
   <JDBCUrl>jdbc:hive2://myhost:10000/default</JDBCUrl>
```

```
        <KeyTabFile>/etc/security/keytabs/hive.keytab</KeyTabFile>
        <PrincipalName>hive/_Host@realm.com</PrincipalName>
    </HiveConfiguration>
```

4. Save the configuration file.

Note: Run the `kinit` command with the same keytab file name and service principal name that you specify for HBase in the configuration file before you run a batch job.

Configuring Relate 360 to Use OpenAM-Based Authentication

You can use the OpenAM-based authentication system to secure the RESTful web services. OpenAM is an open-source access management server that can connect to multiple identity repositories and provide authentication service. The identity repositories can be LDAP directories, relational databases, and Windows authentication. You can configure OpenAM to connect to an identity repository based on the authentication infrastructure of your environment. For example, you can configure OpenAM to connect to Kerberos Key Distribution Center (KDC) to use Kerberos authentication.

1. Deploy OpenAM on an application server.
2. Create a properties file named `security.properties`, and configure the parameters related to OpenAM.

Step 1. Deploy and Configure OpenAM

You must download OpenAM from the OpenAM website and deploy it on an application server. After you deploy OpenAM, configure it to connect to an identity repository based on the authentication infrastructure of your environment.

1. Download the WAR file from the OpenAM website.
2. Deploy OpenAM on a supported application server.
3. Configure OpenAM to connect to a supported identity repository that your environment uses.

For more information about deploying and configuring OpenAM, see the OpenAM documentation.

Step 2. Create the Properties File

After you configure OpenAM, you must create a properties file named `security.properties`, and configure the parameters that are related to OpenAM.

1. Using a text editor, open the following sample file: `/usr/local/mdmbdrm-<Version Number>/sample/security.properties`
2. Configure the following parameters:

   com.informatica.mdmbde.openam.uri.token

   Uniform Resource Identifier (URI) of the OpenAM web service that authenticates the user credentials and generates an authentication token.

Use the following format for the URI:

```
http://<Host Name>:<Port>/openam/json/authenticate?realm=/<Realm Name>
```

The URI uses the following parameters:

- Host Name. Host name of the machine on which you deploy OpenAM.
- Port. Port number on which the host listens.
- Realm Name. Name of the realm that you create in OpenAM.

For example, `http://server99:3333/openam/json/authenticate?realm=/mdmbdrm`

com.informatica.mdmbde.openam.uri.validate

URI of the OpenAM web service that validates the authentication token specified in the web service request.

Use the following format for the URI:

```
http://<Host Name>:<Port>/openam/json/sessions/
```

The URI uses the following parameters:

- Host Name. Host name of the machine on which you deploy OpenAM.
- Port. Port number on which the host listens.

For example, `http://server99:3333/openam/json/sessions/`

com.informatica.mdmdbde.openam.principal

Service Principal Name (SPN) of the HBase master server.

Use the following format for a SPN:

```
http/<Host Name>@<Realm>
```

A SPN uses the following parameters:

- Host Name. Host name of the HBase master server.
- Realm. Authentication administrative domain.

For example, `http/production.domain.com@domain.com`

com.sun.jersey.spi.container.ContainerRequestFilters

Name of the interface that the container request filters use. You must use `com.informatica.mdmbde.rest.security.AuthenticationFilter` as the parameter value.

3. Save the file.

CHAPTER 4

Setting Up the Environment to Process Streaming Data

This chapter includes the following topics:

Processing Streaming Data Overview

You can link or tokenize the streaming data. You can also consolidate the linked data to get the preferred records. Use the JSON format to stream the input data.

Relate 360 uses Apache Kafka and Apache Spark to process the streaming data and the Kerberos authentication to secure the streaming data. Kafka is a distributed messaging system that manages the input data stream. Spark is a distributed realtime computation system that processes the streaming data.

Setting Up the Environment to Process Streaming Data

You must add parameters related to Kafka and the processing mode in the configuration file. You must configure Spark to process the input data.

1. Configure the required parameters in the configuration file.
2. Based on your infrastructure, configure Spark.

Configuring the Required Parameters in the Configuration File

You must configure the parameters related to Kafka in the configuration file. You can also configure the processing mode that indicates whether you want to link or tokenize the input data.

1. Open the configuration file in a text editor.
2. To configure Kafka, configure the following parameters within the `RealTimeService` section:

   Broker

   Host name of the machine that hosts Kafka and the port number on which Kafka listens. If you use a Kafka cluster, you can specify multiple host names and port numbers separated by commas.

   Use the following format to specify the host name and the port number:

   ```
   <Host Name1>:<Port1>,<Host Name2>:<Port2>,...<Host NameN>:<PortN>
   ```

   For example, `KafkaBroker2:9092,KafkaBroker1:9092`

   TopicName

   Name of the Kafka topic to which you want to publish the input data.

   KafkaPrincipalName

   Required if you use Kerberos for authentication. Kerberos principal name of the Kafka broker on which you plan to deploy the Spark instance.

   Use the following format for the Kerberos principal name:

   ```
   kafka/<Full name of the Kafka broker host>@<Realm>
   ```

   For example, `kafka/kafka1.hostname.com@EXAMPLE.COM`

   KafkaKeyTabFile

   Required if you use Kerberos for authentication. Absolute path and file name of the keytab file that contains the Kerberos principal name you specify.

   For example, `/etc/security/keytabs/kafka_server.keytab`

   The directory that contains the keytab file must not be SELinux enabled. To remove the SELinux permissions from a directory, use the `setfattr` command.

   Note: In a distributed Hadoop cluster, ensure that the name of the keytab file and the absolute path to the file are the same for all the nodes.
3. To configure the processing mode, set the `ALTERNATETABLEFORGROUPINFO` property within the `MetaData` section.

   Use one of the following values:

   - True. Indicates to perform the linking process.
   - False. Indicates to perform the tokenization process. Default is false.
4. Save the file.

For example, the following sample code contains the Kafka parameters:

```
<RealTimeService>
   <Broker>localhost:9092</Broker>
   <TopicName>test-22</TopicName>
   <KafkaKeyTabFile>/etc/security/keytabs/kafka_server.keytab</KafkaKeyTabFile>
   <KafkaPrincipalName>kafka/kafka1.hostname.com@EXAMPLE.COM</KafkaPrincipalName>
</RealTimeService>
```

Configuring Spark

You must configure Spark to process the input data. If you use Kerberos for authentication, ensure that the Spark instance runs on YARN.

1. Create symbolic links for the required library files in all the Spark nodes.
2. Configure the environment variables.
3. Run the setup script to deploy Relate 360 on Spark.

Step 1. Create Symbolic Links for the Library Files in the Spark Nodes

Create symbolic links for the required library files in all the Spark nodes and copy the population files, the configuration file, and the matching rules file to all the Spark nodes.

1. Copy the following `.so` files to a Spark node:
    - `/usr/local/mdmbdrm-<Version Number>/bin/libjssan3cl.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libjssan3clex.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libssaiok.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libssalsn.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libssan3cl.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libssan3tb.so`
    - `/usr/local/mdmbdrm-<Version Number>/bin/libssan3v2.so`
2. Create symbolic links to all the `.so` files in the following directory of the Spark node: `$HADOOP_HOME/lib/native`

    The following sample command creates symbolic links in the `$HADOOP_HOME/lib/native` directory for all the `.so` files:

    ```
    ln -s /usr/local/libraryfiles/*.so /usr/lib/hadoop/lib/native/
    ```
3. Copy the population files to the same directory that you configured for the `ssapr` parameter of the configuration file in the Spark node.

    For example, if `ssapr=/usr/local/mdmbdrm-10.0/population`, create the same directory structure in the Spark node and copy the population files to the `population` directory.
4. Copy the configuration file and the matching rules file to the Spark node.

    Ensure that you use the same directory structure in all the Spark nodes.

Step 2. Configure the Environment Variables

Before you set up Spark, you must configure the following environment variables on the machine that has Relate 360 installed:

KAFKA_HOME

Directory in which Kafka is installed.

For example, `export KAFKA_HOME=/opt/cloudera/parcels/KAFKA/lib/kafka`

HBASE_HOME

Directory in which HBase is installed.

For example, `export HBASE_HOME=/usr/hdp/<version>/hbase/`

SPARK_KAFKA_VERSION

Version number of the Kafka consumer API that integrates Spark and Kafka. The supported version number is `0.10`.

SPARK_MAJOR_VERSION

Required only if you use Hortonworks HDP. Major version number of Spark. The supported version number is `2`.

KAFKA_OPTS

Required only if you use Kerberos for authentication. Absolute path to the `jaas.conf` and `krb5.conf` files. In the `jaas.conf` file, ensure that you configure the Kerberos principal name of the Kafka broker and the absolute path to the keytab file.

For example, `export KAFKA_OPTS="-Djava.security.auth.login.config=/data/r360/jaas.conf -Djava.security.krb5.conf=/etc/krb5.conf"`

Note: The directory that contains the `jaas.conf` and `krb5.conf` files must not be SELinux enabled. To remove the SELinux permissions from a directory, use the `setfattr` command.

Note: After you configure the environment variables, ensure that the Kafka broker runs.

Step 3. Deploy Relate 360 on Spark

Deploy Relate 360 on Spark to link, consolidate, or tokenize the input data. To deploy Relate 360 on Spark, run the `setup_realtime.sh` script located in the following directory: `/usr/local/mdmbdrm-<Version Number>`

Note: If you use Cloudera CDH, ensure that you set the `spark2-submit` command as the default command to run the Spark applications.

Use the following command to run the `setup_realtime.sh` script:

```
setup_realtime.sh

--config=configuration_file_name

--rule=matching_rules_file_name

[--checkpointDirectory=directory_to_store_checkpoint]

[--consolidate=consolidation_rules_file_name]

[--driverMemory=driver_memory]

[--enableBackPressure]

[--executorMemory=executor_memory]

[--instanceName=instance_name]

[--keytab=keytab_file_name]

[--maxInputRatePerPartition=number_of_records_per_partition]

[--outputTopic=output_topic_name]

[--probableMatchTopic=probable_match_topic_name]

[--partitions=number_of_partitions]

[--principal=kerberos_principal_name]

[--replica=number_of_replicas]
```

```
[--resumeFrom=checkpoint_value]

[--skipCreateTopic]

[--sparkAppJars=list_of_application_jars]

[--sparkDriverJar=driver_jar]

[--sparkMaster=deployment_mode]

[--sparkMicroBatchDuration=batch_duration]

[--sparkNumCoresPerExecutor=number_of_cores]

[--sparkNumExecutors=number_of_executors]

[--zookeeper=zookeeper_connection_string]
```

The following table describes the options and arguments that you can specify to run the `setup_realtime.sh` script:

Option	Argument	Description
--config	configuration_file_name	Absolute path and file name of the configuration file. Ensure that the configuration file is present in the same directory path in all the Spark nodes.
--rule	matching_rules_file_name	Absolute path and file name of the matching rules file. The values in the matching rules file override the values in the configuration file. Ensure that the matching rules file is present in the same directory path in all the Spark nodes.
--checkpointDirectory	directory_to_store_checkpoint	Optional. Absolute path to a HDFS directory or a shared NFS directory to store the checkpoint-related information. The checkpoint-related information is useful when you redeploy an existing Spark instance after a failure. For example, the following sample directory path stores the checkpoint-related information in HDFS: `hdfs:///user/spark/checkpoint` When you redeploy an existing Spark instance from the failure point, the `setup_realtime.sh` script ignores other options that you specify in the `setup_realtime.sh` script. If you do not want to recover the Spark instance from the failure point, delete the checkpoint directory before you run the `setup_realtime.sh` script or use an unique name for the Spark instance.
--consolidate	consolidation_rules_file_name	Optional. Absolute path and file name of the consolidation rules file. Use the consolidation rules file only when you want to consolidate the linked data and create preferred records for all the clusters.

Option	Argument	Description
--driverMemory	driver_memory	Optional. Amount of memory in gigabytes that you want to allocate to the driver process of the Spark instance. Default is 1g.
--enableBackPressure		Optional. Indicates to enable the internal backpressure mechanism of Spark. The mechanism controls the receiving rate of the streaming data. By default, the internal backpressure mechanism is disabled.
--executorMemory	executor_memory	Optional. Amount of memory in gigabytes that you want to allocate to each executor process of the Spark instance. Default is 1g.
--instanceName	instance_name	Optional. Name for the Spark instance that processes the input data. Default is `BDRMRTIngestSpark`. If you use an existing Spark instance name and specify the check point directory, the `setup_realtime.sh` script ignores other options that you specify when you run the `setup_realtime.sh` script. If you want the `setup_realtime.sh` script to use all the options that you specify, use an unique name for the Spark instance.
--keytab	keytab_file_name	Required if you use Kerberos for authentication. Absolute path and file name of the keytab file. The keytab file must contain the Kerberos principal name that you specify in the `--principal` parameter. The directory that contains the keytab file must not be SELinux enabled. To remove the SELinux permissions from a directory, use the `setfattr` command.
--maxInputRatePerPartition	number_of_records_per_partition	Optional. Maximum number of records that the Spark instance can read from each Kafka partition. By default, the Spark instance reads all the records.
--outputTopic	output_topic_name	Optional. Name of the topic in Kafka to which you want to publish the matching records. By default, the matching records are not published. The script does not create the output topic, so ensure that you create the output topic to publish the matching records to it.
--probableMatchTopic	probable_match_topic_name	Required if you specify a lower threshold score in the configuration file. Name of the topic in Kafka to which you want to publish the probable matching records. The script does not create the probable match topic, so ensure that you create the probable match topic to publish the probable matching records to it.

Option	Argument	Description
--partitions	number_of_partitions	Optional. Number of partitions for the topic. Use partitions to split the data in the topic across multiple brokers. Default is 1. **Note:** Ensure that the number of partitions is equal to the number of node managers in the cluster.
--principal	kerberos_principal_name	Required if you use Kerberos for authentication. Kerberos principal name that has access to submit a Spark job.
--replica	number_of_replicas	Optional. Number of replicas that you want to create for the topic. Use replicas for high availability purposes. Default is 1.
--resumeFrom	checkpoint_value	Optional. Indicates the offset position of the Kafka input topic from which the Spark instance must process the records. Applicable only if you use an unique name in the `--instanceName` option. Configure one of the following values: - `smallest`. Processes all the records from the beginning of the Kafka input topic. - `largest`. Processes the records from the latest position of the Kafka input topic. - `<Instance Name>`. Processes the records based on the offset position of the specified Spark instance that uses the same Kafka input topic. When you redeploy an existing Spark instance after a failure, the Spark instance processes the records in the Kafka topic from the point of failure by default. If you want to process the records from the beginning of the topic or from the current position, reset the offset position for the Spark instance before you redeploy it. For more information about resetting the offset position for a Spark instance, see "Resetting the Offset Position for a Spark Instance" on page 72.
--skipCreateTopic		Required if the topic that you specify in the configuration file already exists in Kafka. Indicates to skip creating the topic. By default, the script creates the topic.

Option	Argument	Description
--sparkAppJars	list_of_application_jars	Optional. Comma-separated list of library JAR files and their paths that you want to include in the driver and executor class paths. You can specify the following JAR files: - `/usr/local/mdmbdrm-<Version Number>/bin/BDRMRTProcessor.jar` - `/usr/local/mdmbdrm-<Version Number>/bin/fastutil-7.0.2.jar` - `/usr/local/mdmbdrm-<Version Number>/bin/guava-12.0.1.jar` - `/usr/local/mdmbdrm-<Version Number>/bin/htrace-core.jar` - `/usr/local/mdmbdrm-<Version Number>/bin/ssan3.jar`
--sparkDriverJar	driver_jar	Optional. Name and path of the `bdrm-rt-ingest-spark-10.0.HF5.jar` file to include in the driver and executor class paths. You can find the `bdrm-rt-ingest-spark-10.0.HF5.jar` file in the following directory: `/usr/local/mdmbdrm-<Version Number>/bin`
--sparkMaster	deployment_mode	Indicates whether the Spark runs in the standalone or cluster mode. Use one of the following values: - local[*]. Indicates to run Spark locally with as many worker threads as the number of cores on your machine. - local[N]. Indicates to run Spark locally with N worker threads. - yarn. Indicates to run Spark in the cluster mode. Default is `local[*]`.
--sparkMicroBatchDuration	batch_duration	Optional. Number of seconds for the Spark instance to wait before packaging the input records into a batch. Default is 2. **Note:** When you kill a batch in the Spark web UI, the Spark instance skips the unprocessed records in the batch.
--sparkNumCoresPerExecutor	number_of_cores	Optional. Number of cores for each executor process to use. Default is 1.

Option	Argument	Description
--sparkNumExecutors	number_of_executors	Optional. Number of executor processes that you want to use for the Spark instance. By default, the number of executor processes depends on the data size and the number of node managers in the cluster. Applicable only when you run the Spark instance on YARN.

--zookeeper	zookeeper_connection_string	Connection string to access the ZooKeeper server.
		Use the following format for the connection string:
		`<Host Name>:<Port>[/<chroot>]`
		The connection string uses the following parameters:
		- `Host Name`. Host name of the ZooKeeper server.
		- `Port`. Port on which the ZooKeeper server listens.
		- `chroot`. Optional. ZooKeeper root directory that you configure in Kafka. Default is /.
		The following example connection string uses the default ZooKeeper root directory:
		`server1.domain.com:2182`
		The following example connection string uses the user-defined ZooKeeper root directory:
		`server1.domain.com:2182/kafkaroot`
		If you use an ensemble of ZooKeeper servers, you can specify multiple ZooKeeper servers separated by commas.

For example, the following command runs the script that deploys Relate 360 on Spark:

```
setup_realtime.sh --config=/usr/local/conf/config_big.xml --rule=/usr/local/conf/
matching_rules.xml --resumeFrom=smallest --instanceName=Prospects --
zookeeper=10.28.10.345 --partitions=3 --replica=2 --sparkMaster=yarn --
sparkMicroBatchDuration=5 --checkpointDirectory=hdfs:///user/spark/checkpoint --
outputTopic=InsuranceOutput --driverMemory=2g --executorMemory=2g --sparkNumExecutors=3
--sparkNumCoresPerExecutor=2 --sparkAppJars=$sparkDriverLibraryPath/ssan3.jar,
$sparkDriverLibraryPath/BDRMRTProcessor.jar,$sparkDriverLibraryPath/fastutil-7.0.2.jar,
$sparkDriverLibraryPath/htrace-core.jar,$sparkDriverLibraryPath/guava-12.0.1.jar --
sparkDriverJar=$sparkDriverLibraryPath/bdrm-rt-ingest-spark-10.0.HF5.jar --
maxInputRatePerPartition=40 --sparkMicroBatchDuration=10 --enableBackPressure --
principal=kafka/kafka1.hostname.com@EXAMPLE.COM --keytab=/etc/security/keytabs/
kafka_server.keytab --probableMatchTopic=probableOutput
```

Resetting the Offset Position for a Spark Instance

When you redeploy an existing Spark instance after a failure, the Spark instance processes the records in the Kafka input topic from the point of failure by default. If you want to process the records from the beginning of the topic or from the latest position of the topic, reset the offset position for the Spark instance before you redeploy it.

To reset the offset position, use the `kafka-consumer-groups.sh` script located in the following directory: `$KAFKA_HOME/bin`

To run the `kafka-consumer-groups.sh` script, use the following command:

```
kafka-consumer-groups.sh

--bootstrap-server <list_of_kafka_brokers_with_ports>

--group <consumer_group_name>

--topic <kafka_input_topic>

--reset-offsets

--to-earliest|--to-latest

--execute
```

The following table describes the options and arguments that you can specify to run the `kafka-consumer-groups.sh` script:

Option	**Argument**	**Description**
--bootstrap-server	list_of_kafka_brokers_with_ports	Comma-separated list of Kafka brokers and their port numbers. For example, `broker1:9092,broker2:9092`.
--group	consumer_group_name	Name of the Kafka consumer group. The name is based on the Spark instance name and uses the following format: `<Spark instance Name>-spark-bdrm` For example, if the instance name is `BDRMRTIngestSpark`, then the consumer group name is `BDRMRTIngestSpark-spark-bdrm`.
--topic	kafka_input_topic	Name of the topic to which Kafka publishes the input data stream.
--reset-offsets		Resets the offset position for the Spark instance.
--to-earliest | --to-latest		Indicates the offset position to set. Use one of the following options: - `--to-earliest`. Resets the offset position to the beginning of the topic. - `--to-latest`. Resets the offset position to the latest position of the topic.
--execute		Runs the script.

After you successfully run the `kafka-consumer-groups.sh` script, redeploy the Spark instance.

RELATED TOPICS:

- “Step 3. Deploy Relate 360 on Spark” on page 67

Troubleshooting Spark Deployment

If you encounter any issues when you deploy Spark on Relate 360, use the following information to troubleshoot the issues.

The ingestion of streaming data failed with too many ZooKeeper connections error.

When an ingestion request uses multithreaded processing and exceeds the allowed number of concurrent ZooKeeper connections, the ingestion of streaming data fails with the following error message:

```
Too many connections from <Client> - max is 60
```

Default is 60 connections. To fix this issue, increase the allowed number of concurrent ZooKeeper connections. To configure the allowed number of concurrent ZooKeeper connections, use the `maxClientCnxns` property in the ZooKeeper configuration file.

CHAPTER 5

Configuring Distributed Search

This chapter includes the following topics:

Distributed Search Overview

Distributed search is a search model that distributes a search request to multiple region servers of the repository and performs the search on the region servers. Distributed search optimally uses the available resources and results in improved search performance. Distributed search uses HBase coprocessors that run on the region servers to perform the searches.

Configuring Distributed Search

You must configure distributed search before you load the linked or tokenized data to the repository for the first time.

1. Create symbolic links for the required library files in the region servers.
2. Add the distributed search-related parameters to the configuration file.
3. Package and deploy a coprocessor JAR file for the region servers.

Creating Symbolic Links for the Library Files in the Region Servers

You must copy the required library files to each region server that stores data. You must then create symbolic links for the copied library files in the appropriate directories of the region servers so that the region servers can perform the searches.

1. Copy the following `.so` and `.JAR` files to a region server:
   - `/usr/local/mdmbdrm-<Version Number>/libjssan3clex.so`
   - `/usr/local/mdmbdrm-<Version Number>/libssan3v2.so`
   - `/usr/local/mdmbdrm-<Version Number>/libssan3tb.so`

- `/usr/local/mdmbdrm-<Version Number>/libssan3cl.so`
- `/usr/local/mdmbdrm-<Version Number>/libssalsn.so`
- `/usr/local/mdmbdrm-<Version Number>/libssaiok.so`
- `/usr/local/mdmbdrm-<Version Number>/libjssan3cl.so`
- `/usr/local/mdmbdrm-<Version Number>/bin/ssan3.jar`

2. Create symbolic links for the `.so` files in the following directory of the region server: `$HADOOP_HOME/lib/native`

   The following sample command creates symbolic links in the `$HADOOP_HOME/lib/native` directory for all the `.so` files:

   ```
   ln -s /usr/local/mdmbdrm-10.0/bin/*.so /usr/lib/hadoop/lib/native/
   ```

3. Create symbolic link for the `ssan3.jar` file in the following directory of the region server: `$HBASE_HOME/lib/`

   The following sample command creates symbolic link in the `$HBASE_HOME/lib/` directory for the `ssan3.jar` file:

   ```
   ln -s /usr/local/mdmbdrm-10.0/bin/ssan3.jar /usr/lib/hbase/lib
   ```

4. Copy the population files to the same directory that you configured for the `ssapr` parameter of the configuration file in the region server.

   For example, if `ssapr=/usr/local/mdmbdrm-10.0/population`, create the same directory structure in the region server and copy the population files to the `population` directory.

Adding the Distributed Search-Related Parameters to the Configuration File

Before you generate the coprocessor JAR file, you must update the configuration file to include the parameters related to distributed search.

1. Open the configuration file in a text editor.

2. Add the following parameters within the `HBASEConfiguration` section:

   CoprocessorPath

   Absolute path and file name for the coprocessor JAR file that you must generate and deploy in HDFS. The JAR file contains the search logic that the region servers use to perform searches. For example, `/user/cloudera/db-hbase-coprocessorDeploy.jar`.

   Use the following name format for the JAR file name: `db-hbase-coprocessor<id>`

   The name format uses the `id` parameter that indicates a unique identifier for the JAR file. For example, if `id=Deploy`, the JAR file name must be `db-hbase-coprocessorDeploy.jar`.

   CoprocessorClass

   Name of the class that the coprocessor uses. Specify `com.informatica.mdmbde.database.hbase.coprocessor.BDRMRegionObserver` as the parameter value.

   ScanCacheSize

   Optional. Number of records to pass to scanners at once. Adjust the value of the `ScanCacheSize` parameter based on the `ScanMaxResultSize` parameter.

   A lower value can result in multiple scans that might affect the performance of the jobs. A higher value can result in the failure of jobs if the size of the records exceeds the value of the `ScanMaxResultSize` parameter.

   Default is 500.

   ScanMaxResultSize

   Optional. Maximum size of records in bytes to return to scanners at once. Adjust the value of the `ScanMaxResultSize` parameter based on the `ScanCacheSize` parameter.

   A lower value can result in multiple scans and a higher value can result in increased memory usage that might affect the performance of the jobs.

   Default is 2097152.

Packaging and Deploying the Coprocessor JAR File

For the region servers to perform searches, you must package the search logic in a JAR file. You can then deploy the generated JAR file to a directory in HDFS so that the region servers can access the search logic.

To package the search logic in a JAR file, run the `run_deployment.sh` script located in the following directory: `/usr/local/mdmbdrm-<Version Number>`

1. Use the following command to run the `run_deployment.sh` command:

```
run_deployment.sh

--config=configuration_file_name

--installbin=install_bin_directory

--coprocessor

--id=identifier_for_the_JAR_file

[--rule=matching_rules_file_name]
```

The following table describes the options and arguments that you must specify to run the `run_deployment.sh` script:

Option	Argument	Description
--config	configuration_file_name	Absolute path and file name of the configuration file that you create.
--installbin	install_bin_directory	Absolute path to the `bin` directory of the Relate 360 installation.
--id	identifier_for_the_JAR_file	Unique identifier for the coprocessor JAR file. The name format of the generated JAR file is `db-hbase-coprocessor<id>`. For example, if `id=Deploy`, the generated JAR file name is `db-hbase-coprocessorDeploy.jar`. Ensure that you use the same identifier that you used in the `CoprocessorPath` parameter of the configuration file.
--rule	matching_rules_file_name	Optional. Absolute path and file name of the matching rules file that you create.
--coprocessor		Indicates that the script generates the coprocessor JAR file.

2. Deploy the generated JAR file to the directory that you configured for the `CoprocessorPath` parameter in the configuration file.

   For example, the following command deploys a JAR file to an HDFS directory:

   ```
   hadoop fs -put -f db-hbase-coprocessorDeploy.jar /user/cloudera/
   ```

CHAPTER 6

Packaging and Deploying the RESTful Web Services

This chapter includes the following topics:

RESTful Web Services Overview

The RESTful web services are based on the REST architecture. You can use the RESTful web services to search records, to get a list of records in a cluster, and to ingest streaming data.

Before you use the RESTful web services, you must package them in a web application archive (WAR) file, which is a packaged JAR file. When you package the RESTful web services, you can also package the relationship graph user interface in another WAR file. After you generate the WAR files, deploy them on a Tomcat container.

Packaging the RESTful Web Services

Before you use the RESTful web services, package the web services in a WAR file. You can use the configuration file, the matching rules file, and the relationship configuration file to package the web services. If you want to package the relationship graph user interface, you must specify the properties file for the relationship graph.

Run the `run_deployment.sh` script located in the following directory to package the web services in a WAR file: `/usr/local/mdmbdrm-<Version Number>`

The `run_deployment.sh` script uses one of the following pre-installed WAR files based on the HBase version that you use:

- For HBase version 0.94.x, `/usr/local/mdmbdrm-<Version Number>/bin/MDMBDRMV94.war`

  Note: You must rename the `MDMBDRMV94.war` file to `MDMBDRM.war` before you run the `run_deployment.sh` script.

- For HBase version 0.96.x or later, `/usr/local/mdmbdrm-<Version Number>/bin/MDMBDRM.war`

To run the `run_deployment.sh` script, use the following command:

```
run_deployment.sh

--config=configuration_file_name

--rule=matching_rules_file_name

--installbin=install_bin_directory

[--relationship=relationship_configuration_file_name]

[--uiconfig=ui_properties_file]

[--override]

[--id=configuration_matching_rules_files_identifier]

[--security=security_properties_file_name]
```

The following table describes the options and arguments that you must specify to run the `run_deployment.sh` script:

Option	Argument	Description
--config	configuration_file_name	Absolute path and file name of the configuration file that you create.
--rule	matching_rules_file_name	Optional. Absolute path and file name of the matching rules file that you create.
--installbin	install_bin_directory	Absolute path to the `bin` directory of the Relate 360 installation.
--relationship	relationship_configuration_file_name	Optional. Absolute path and file name of the relationship configuration file. If you plan to create relationships between the processed data, you must specify the relationship configuration file.
--uiconfig	ui_properties_file	Optional. Absolute path and file name of the properties file that you create for the relationship graph user interface. If you want to package the relationship graph user interface in a WAR file, specify the properties file. You require only one relationship graph user interface WAR file for your environment. The default package name for the relationship graph user interface is `bdrm-ui.war`.

Option	Argument	Description
--override		Optional. Indicates to include the generated JAR file into the `MDMBDRM.war` file. If you want to use multiple configuration files, you can specify the `--override` option to override the `MDMBDRM.war` file. **Note:** You must have the root privileges to override the `MDMBDRM.war` file. By default, the script loads the generated WAR file in the local working directory.
--id	configuration_matching_rules_files_identifier	Optional. Unique identifier for the configuration file and the matching rules file. The WAR file generated in the local directory and the REST API URLs use the identifier. The name format of the WAR file generated in the local directory is `MDMBDRM<id>`, and the format of the REST API URL is `http://<Host>:<Port>/MDMBDRM<id>/<Version Number>/<id>/<Operation Name>`. For example, if you specify `--id=Dev20`, the name of the WAR file generated in the local directory is `MDMBDRMDev20.war`, and the format of the REST API URL is `http://<Host>:<Port>/MDMBDRMDev20/<Version Number>/Dev20/<Operation Name>`. By default, the WAR file generated in the local directory and the REST API URLs use a system-generated identifier, which is difficult to interpret. For example, `http://localhost:8080/MDMBDRM1131329172862026293/v5.0/0fb349e3-807a-4235-a100-134a3b51ad5d`.
--security	security_properties_file_name	Required if you configure security for the web services. Absolute path and name of the `security.properties` file.

For example, the following sample command generates a WAR file for the RESTful web services and another WAR file for the relationship graph user interface:

```
run_deployment.sh --config=/usr/local/conf/config_big.xml --rule=/usr/local/conf/
matching_rules.xml --relationship=/usr/local/conf/relconfig.xml --installbin=/usr/local/
mdmbdrm-10.0/bin --id=Customer360 --security=/usr/local/conf/security.properties --
uiconfig=/usr/local/conf/ui.properties
```

Deploying the RESTful Web Services

After you package the web services and the relationship graph user interface, deploy the packages on a Tomcat container. In a distributed Hadoop cluster, if you have configured security, ensure that the Tomcat is deployed within the Hadoop cluster.

1. Copy the WAR file that contains the RESTful web services to the following directory: `$CATALINA_HOME/webapps`.
2. If you want to deploy the relationship graph user interface, copy the `bdrm-ui.war` file to the following directory: `$CATALINA_HOME/webapps`.
3. Copy the `/usr/local/mdmbdrm-<Version Number>/bin/ssan3.jar` file to the following directory: `$CATALINA_HOME/lib`
4. Export the `LD_LIBRARY_PATH` environment variable to include the following directory: `/usr/local/mdmbdrm-<Version Number>/bin` directory.

   For example, `export LD_LIBRARY_PATH=/usr/local/mdmbdrm-<Version Number>/bin`
5. Export the `JAVA_OPTS` environment variable to configure the minimum and maximum heap sizes based on the amount of available memory.

   For example, `export JAVA_OPTS="-Xms1536m -Xmx1536m"`
6. To start the Tomcat server, from the `bin` directory of the Tomcat installation, run the following command:

   ```
   ./startup.sh
   ```
7. To view the list of RESTful web services that you deploy, type the following URL format in a browser:

   ```
   http://<Host>:<Port>/<WAR File Name>
   ```

   For example, `http://BDRMServer:8080/MDMBDRMCustomer360`

   The following sample shows a list of RESTful web services that are deployed based on a WAR file named `MDMBDRMCustomer360`:

   ```
   [
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/SEARCH",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETCLUSTER",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETSEARCHLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETCLUSTERLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETMULTISEARCHLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/MULTISEARCH",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/RULESEARCH",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/INGEST",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETINGESTLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETRECORDLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETRECORD",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/
   GETMANAGECLUSTERLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/MANAGECLUSTER",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/DELETERECORD",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/AUTHENTICATE",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/
   GETPREFERREDRECORDLAYOUT",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETPREFERREDRECORD",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETSTRATEGIES",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/
   PREFERREDRECORDSEARCH",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETENTITYMETADATA",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETENTITYDETAILS",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/
   GETENTITYRELATIONSHIP",
      "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETGRAPHMETADATA",
   ```

```
        "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/GETALLRELATIONSHIPS",
        "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/CREATERELATIONSHIP",
        "http://BDRMServer:8080/MDMBDRMCustomer360/v5.0/Customer360/REMOVERELATIONSHIP"
    ]
```

Note: If you secure the RESTful web services, run the AUTHENTICATE web service request to authenticate the user credentials and then type the URL in the browser.

For more information about the AUTHENTICATE web service, see the *Informatica MDM - Relate 360 User Guide*.

CHAPTER 7

Troubleshooting

This chapter includes the following topic:

Troubleshooting Search

If you encounter any issues when you run search requests, use the following information to troubleshoot.

In a distributed search environment, a search request returns a PartialRowException error.

A search might fail in a distributed search environment when you have a mismatch between the `ScanMaxResultSize` and `ScanCacheSize` parameter values.

To fix this issue, adjust the values of the `ScanMaxResultSize` and `ScanCacheSize` parameters so that the size of the records that the `ScanCacheSize` parameter value returns is less than the `ScanMaxResultSize` parameter value.

INDEX