Informatica

Informatica® PowerExchange for Cassandra JDBC

10.2.2

User Guide

Informatica PowerExchange for Cassandra JDBC User Guide
10.2.2
February 2019

Publication Date: 2019-02-20

Table of Contents

Preface

The *Informatica PowerExchange for Cassandra JDBC User Guide* describes how to use PowerExchange for Cassandra with Informatica Data Services to extract data from and load data to Cassandra. The guide is written for database administrators and developers who are responsible for developing mappings and workflows. This guide assumes that you have knowledge of Cassandra and Informatica Data Services

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

Introduction to PowerExchange for Cassandra JDBC

This chapter includes the following topics:

PowerExchange for Cassandra JDBC Overview

Use PowerExchange for Cassandra JDBC to connect to a Cassandra database using the Informatica Cassandra JDBC driver. You can read data from or write data to column families in a Cassandra keyspace through the Data Integration Service.

PowerExchange for Cassandra JDBC supports Cassandra Query Language (CQL) to write queries to the Cassandra database. The Cassandra JDBC driver creates virtual tables to renormalize the data in Cassandra collections, such as a list, set, or map.

You can use PowerExchange for Cassandra JDBC in the following data integration scenarios:

- Your organization needs to extract data from Cassandra and aggregate data into a data warehouse or data mart for interactive data exploration and analysis.
- Your organization needs to load data from flat files, relational database, or message queues to Cassandra.
- Your organization needs to ensure consistency between your SQL data store and Cassandra data store.

Introduction to Cassandra

Cassandra is an open source, NoSQL database that is highly scalable and provides high availability. You can use Cassandra to store large amounts of data spread across data centers or when your applications require high write access speed.

In a Cassandra database, a column family is similar to a table in a relational database and consists of columns and rows. Similar to the relational database, each row is uniquely identified by a row key. The column name uniquely identifies each column in the column family. The number of columns in each row can vary, and client applications can determine the number of columns in each row.

You can read, write, and manipulate a group of data by using collections in Cassandra. The Cassandra database supports the following collection types:

- Set
- List
- Map

To effectively query data, you can use the dynamic column family feature in Cassandra. For example, the following CQL definition creates a column family that can store information about users and their friend lists.

```
CREATE TABLE users_list(
    username text PRIMARY KEY,
    friendlist list<text>);
```

Each row in the users_list column family contains a user name and the corresponding list of friends for each user. The friendlist column is a collection of list type. The rows in the users_list column family can contain different number of friendlist columns, and each row effectively represents a snapshot of a query on a user's friend list.

The following CQL insert statement inserts a row that contains a user name and the corresponding friend list:

```
insert into users_list (username, friendlist) values(user1,{' userxyz', 'userabc','
userqwe'});
```

Virtual Tables

PowerExchange for Cassandra JDBC creates virtual tables if the column family contains collections such as set, list, or map.

Virtual tables depict the renormalized view of a Cassandra collection. You can import virtual tables as a Cassandra data object and create mappings. PowerExchange for Cassandra JDBC creates a virtual table for each collection in the column family. The virtual table for a collection uses the following naming convention by default: `<original column family name>_vt_<original collection name>`

Each virtual table has a key column that references back to the primary key column in the original column family. The virtual table has an index column that shows the position of the data within the original array. The index column uses the following naming convention by default: `<original column name>_index`. Other columns in the virtual table represent the elements in the array and are named after the array element. If the array is of scalar type, the data column uses the following naming convention by default: `<original column name>_value`

Example

The following CQL statement defines a Cassandra column family that contains a set:

```
CREATE TABLE cloud.users_list (
    username text PRIMARY KEY,
    first_name text,
    last_name text,
    city text,
    last_login timestamp,
    emails set<ascii>
);
```

When you use PowerExchange for Cassandra JDBC to import the column family, the Simba Cassandra JDBC Driver creates the users_list_vt_emails virtual table for the users_list table.

The following table shows the denormalized view of the users_list_vt_emails virtual table:

username	emails_value
user1	xyz@gmail.com
user1	abc@hotmail.com
user1	qwe@yahoo.com
user2	zxc@gmail.com
user2	jkl@hotmail.com

CHAPTER 2

PowerExchange for Cassandra JDBC Configuration

This chapter includes the following topics:

PowerExchange for Cassandra JDBC Configuration Overview

PowerExchange for Cassandra installs with Informatica services.

The Informatica Cassandra JDBC driver is installed on the machines on which you installed Informatica services and clients. PowerExchange for Cassandra JDBC supports the version of the CQL that the Cassandra database supports.

To configure PowerExchange for Cassandra JDBC, complete the prerequisites.

Prerequisites

PowerExchange for Cassandra JDBC installs with the Informatica services and clients.

Before you can use PowerExchange for Cassandra JDBC, perform the following tasks:

- Install or upgrade the Informatica services.
- Create a Data Integration Service and a Model Repository Service in the Informatica domain.
- Ensure that you have the PowerExchange for Cassandra license file.

CHAPTER 3

Cassandra Connections

This chapter includes the following topics:

Cassandra Connections Overview

Use a Cassandra connection to access a Cassandra table.

Use the Cassandra connection to import Cassandra tables, create data objects, preview data, and run mappings. When you create a Cassandra connection, you define the connection attributes that the Developer tool uses to connect to the Cassandra tables.

Use the Developer tool, Administrator tool, or infacmd to create a Cassandra connection.

Cassandra Connection Properties

When you set up a Cassandra connection, you must configure the connection properties.

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

The following table describes the Cassandra connection properties:

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

Property	Description
Description	Optional. The description of the connection. The description cannot exceed 4,000 characters.
Location	The domain where you want to create the connection.
Type	The connection type. Select **Cassandra**.
Host Name	Host name or IP address of the Cassandra server.
Port	Cassandra server port number. Default is 9042.
User Name	User name to access the Cassandra server.
Password	Password corresponding to the user name to access the Cassandra server.
Default Keyspace	Name of the Cassandra keyspace to use by default.
SQL Identifier Character	Type of character that the database uses to enclose delimited identifiers in SQL or CQL queries. The available characters depend on the database type. Select **None** if the database uses regular identifiers. When the Data Integration Service generates SQL or CQL queries, the service does not place delimited characters around any identifiers. Select a character if the database uses delimited identifiers. When the Data Integration Service generates SQL or CQL queries, the service encloses delimited identifiers within this character.
Additional Connection Properties	Enter one or more JDBC connection parameters in the following format: `<param1>=<value>;<param2>=<value>;<param3>=<value>` PowerExchange for Cassandra JDBC supports the following JDBC connection parameters: - BinaryColumnLength - DecimalColumnScale - EnableCaseSensitive - EnableNullInsert - EnablePaging - RowsPerPage - StringColumnLength - VTTableNameSeparator
SSL Mode	Not applicable for PowerExchange for Cassandra JDBC. Select **disabled**.
SSL Truststore Path	Not applicable for PowerExchange for Cassandra JDBC.
SSL Truststore Password	Not applicable for PowerExchange for Cassandra JDBC.
SSL Keystore Path	Not applicable for PowerExchange for Cassandra JDBC.
SSL Keystore Password	Not applicable for PowerExchange for Cassandra JDBC.

Creating an Cassandra Connection

Create a Cassandra connection before you create a Cassandra data object.

1. In the Developer tool, click **Window** > **Preferences**.
2. Select **Informatica** > **Connections**.
3. Expand the domain in the **Available Connections**.
4. Select the connection type **NoSQL** > **Cassandra**, and click **Add**.
5. Enter a connection name and an optional description.
6. Select **Cassandra** as the connection type.
7. Click **Next**.
8. Configure the connection properties.
9. Click **Test Connection** to verify the connection to Cassandra.
10. Click **Finish**.

CHAPTER 4

PowerExchange for Cassandra JDBC Data Objects

This chapter includes the following topics:

Cassandra Data Object Overview

A Cassandra data object is a physical data object that uses Cassandra as a source. A Cassandra data object is a physical data object that represents data based on a Cassandra table.

You can configure the data object read operation properties that determine how the Data Integration Service reads data from Cassandra sources.

Create a Cassandra data object in the Developer tool. PowerExchange for Cassandra JDBC creates the data object read operation for the Cassandra data object. You can edit the advanced properties of the data object read operation and add it to a mapping.

Cassandra Data Object Properties

Specify the data object properties when you create the data object.

The following table describes the properties that you configure for a Cassandra data object:

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

Pre SQL and Post SQL Commands

You can specify **preSQL** and **postSQL** data object read and write operation properties for Cassandra data objects. When you create a data object read or write operation in the Developer tool, you can specify SQL commands on the **Advanced** view.

You can perform the following operations by using pre SQL and post SQL commands:

- INSERT
- UPDATE
- DELETE

Cassandra Data Object Read Operation

Create a mapping with a Cassandra data object read operation to read data from Cassandra.

Output Properties of the Data Object Read Operation

The output properties represent data that the Data Integration Service passes into the mapping pipeline. Select the output properties to configure advanced properties of the data object read operation.

The output properties of the data object read operation include general properties that apply to the data object operation. The output properties also include source, query, run-time, and advanced properties that apply to the Cassandra data object.

You can view and change the output properties of the data object read operation from the **General**, **Sources**, **Query**, **Run-time**, and **Advanced** tabs.

General Properties

The general properties display the name and description of the data object read operation.

Sources Properties

The sources properties list the Cassandra objects used in the data object read operation. You can join data from multiple sources of the Cassandra data object in a read operation.

Query Properties

Use the **Query** tab to select specific records from a Cassandra table.
The following table describes the query properties that you configure for a data object read operation:

Property	Description
Query	Filter value in a read operation. The filter specifies the where clause of select statement. Use a filter to reduce the number of rows that the Data Integration Service reads from the source. When you enter a source filter, the Developer tool adds a WHERE clause to the default query. You can use the Native or Platform expression to select specific records.

Run-time Properties

The run-time properties displays the name of the connection that the Data Integration Service uses to read data from the Cassandra table.

You can define the partition type as key range partitioning to read data from Cassandra data object. When you configure key range partitioning, the Data Integration Service distributes rows of data based on a port or set of ports that you define as the partition key. You can define a range of values for each port. The Data Integration Service uses the key and ranges to send rows to the appropriate partition.

Advanced Properties

The Data Integration Service reads data from Cassandra based on the data object read operation.

The Developer tool displays advanced properties for the Cassandra data object operation in the Advanced view.
The following table describes the advanced properties for a Cassandra data object read operation:

Property	Description
preSQL	SQL statement that you want to run before reading data from the source. For example, if you want to insert records in the database before you read the records from the table, specify the following pre SQL statement: `INSERT INTO cloud.emp (emp_id, emp_city, emp_name, emp_phone, emp_sal) VALUES(7, 'New York','John', 9848022441, 95000);`
postSQL	SQL statement that you want to run after reading data from the source. For example, if you want to delete records in a table after you read the records from a source table, specify the following post SQL statement: `DELETE FROM cloud.emp WHERE emp_id in (7);`

Cassandra Data Object Write Operation

Create a mapping to write data to Cassandra. Use the Cassandra connection, and define the write operation properties to write data to Cassandra.

You can perform insert, update, and upsert operations on a Cassandra target.

Input Properties of the Data Object Write Operation

Input properties represent data that the Data Integration Service writes to a Cassandra table. Select the input properties to configure run-time properties of the data object write operation. You can also specify advanced data object write operation properties to write data to Cassandra table.

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

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

General Properties

The general properties list the name and description of the data object write operation.

Target Properties

The target properties list the Cassandra table in the data object write operation.

Ports Properties - Input Write

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

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

Property	Description
Name	Name of the port.
Type	Data type of the port.
Precision	Maximum number of significant digits for numeric data types, or maximum number of characters for string data types. For numeric data types, precision includes scale.
Scale	Maximum number of digits after the decimal point for numeric values.
Description	Description of the port.

Run-time Properties

The run-time properties displays the name of the connection that the Data Integration Service uses to write data to the Cassandra table.

Advanced Properties

Cassandra data object write operation properties include advanced properties that apply to the Cassandra data object.

The Developer tool displays advanced properties for the Cassandra data object operation in the **Advanced** tab.

You can configure the following advanced properties in the data object write operation:

Property	Description
UpdateMode	Determines the mode that the Data Integration Service uses to update rows in the Cassandra target. You can select one of the following modes: - Update As Update. - Update Else Insert. Default is Update As Update.
Truncate Target Table	The Data Integration Service truncates the Cassandra target before it writes data to the target. Default is selected. If you specify an SQL statement in the Pre SQL property, the Data Integration Service runs the SQL statement before it truncates the table. **Note:** This property is not applicable for Cassandra tables that contain collection data types.
Pre SQL	SQL statement that you want to run before writing data to the target. For example, if you want to insert records from the database before you write the records into the table, specify the following pre SQL statement: `INSERT INTO cloud.emp (emp_id, emp_city, emp_name, emp_phone, emp_sal) VALUES(7, 'New York','John', 9848022441, 95000);`
Post SQL	SQL statement that you want to run after writing the data into the target. For example, if you want to delete records in a table after you write the records into the target table, specify the following post SQL statement: `DELETE FROM cloud.emp WHERE emp_id in (7);`

Creating an Cassandra Data Object

Create a Cassandra data object to add to a mapping.

1. Select a project or folder in the **Object Explorer** view.
2. Click **File** > **New** > **Data Object**.
3. Select **Cassandra Data Object** and click **Next**.

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

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

Creating a Cassandra Data Object Operation

You can create the data object read or write operation for Cassandra data objects. You can the add the Cassandra data object operation to a mapping.

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

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

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

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

Rules and Guidelines for Cassandra Data Object Operations

Use the following rules and guidelines when you configure a Cassandra data object read and write operation:

- You cannot configure a Cassandra data object lookup operation to lookup data from a Cassandra source based on a lookup condition.
- You cannot configure an Update Strategy transformation to write data to a Cassandra table.
- You cannot perform more than one operation with a pre SQL or post SQL command.
- If you specify the host name, port number, user name, and password of the Cassandra server in the connection and **Additional Connection Properties**, the values specified in the **Additional Connection Properties** takes precedence.
- When you read data from or write data to a Cassandra virtual table, you must not specify **1** as the value of the **QueryMode** parameter in the additional connection properties.

CHAPTER 5

PowerExchange for Cassandra JDBC Mappings

This chapter includes the following topics:

PowerExchange for Cassandra JDBC Mappings Overview

After you create a Cassandra data object read or write operation, you can create a mapping to extract data from a Cassandra source or load data to a Cassandra target.

You can define properties in an operation to determine how the Data Integration Service must extract data from a Cassandra source or load data to a Cassandra target. You can extract data from one or more Cassandra sources, and load data to one or more Cassandra targets. When the Data Integration Service extracts data from the source or loads data to the target, it converts the data based on the data types associated with the source or the target.

Mapping Validation and Run-time Environments

You can validate and run mappings on the Spark engine in the Hadoop environment.

The Data Integration Service validates whether the mapping can run in the Hadoop environment.

When you run mappings on the Spark engine, the Data Integration Service pushes the mapping to a Hadoop cluster and processes the mapping on a Spark engine. The Data Integration Service generates an execution plan to run mappings on the Spark engine.

You can view the plan in the Developer tool before you run the mapping and in the Administrator tool after you run the mapping.

For more information about the Hadoop environment and Spark engines, see the *Informatica Big Data Management™ User Guide*.

CHAPTER 6

Virtual Table Operations

This chapter includes the following topics:

Virtual Table Operations Overview

PowerExchange for Cassandra JDBC creates a separate virtual table for each collection type in the column family. When you use a virtual table in a mapping, you can perform read, write, append, update, or delete operations.

Virtual Table Operations

You can use virtual table operations to access, write, or delete rows from a virtual table.

You can perform the following operations in a virtual table:

Read

Use the read operation to retrieve rows from a virtual table.

Insert

Use the insert operation to add one or more rows to a virtual table. Each row that you insert must correspond to a primary key in the source table. If you insert data into a row key that contains data, Cassandra overwrites the original data in the row with the new data.

Note: Cassandra database treats all insert operations as upsert operations.

Update

Use the update operation to update rows in a virtual table.

Note: If the collection type is list, sort the corresponding source or target virtual table data in descending order before you pass data to the Update Strategy transformation.

Virtual Table Operations Examples

The example mappings illustrate the operations that you can perform on a virtual table.

The login details of users are stored in a Cassandra column family. You need to read, insert, and update values from the location column.

The following *users_list* table represents a Cassandra column family that contains a collection of type list:

username	first_name	last_name	city	last_login	location
user1	john	doe	SFO	2014-8-03 01:30:00	0.0.0.0, 123.123.123.2
user2	jane	smith	NYC	2014-8-03 01:40:00	123.88.99.2, 0.0.0.0
user3	brian	lee	SFO	2014-8-04 11:15:00	123.123.2.2, 127.0.0.0
user4	james	adam	NYC	2014-8-05 05:22:00	123.123.1.2, 127.0.0.0

PowerExchange for Cassandra JDBC creates the *users_list* main table and *users_list_vt_location* virtual table.

The following image displays the *users_list* main table:

	Name	Native Type	Precisi...	Scale	Primar...	Nullable	Description
1	username	varchar	4000	0	☑	☐	
2	city	varchar	4000	0	☐	☑	
3	first_name	varchar	4000	0	☐	☑	
4	last_login	timestamp	29	9	☐	☑	
5	last_name	varchar	4000	0	☐	☑	

The following image displays the *users_list_vt_location* virtual table:

Columns

	Name	Native Type	Precisi...	Scale	Primar...	Nullable	Description
1	username	varchar	4000	0	☑	☐	
2	location_index	integer	10	0	☑	☐	
3	location_value	varchar	4000	0	☐	☐	

The following example mappings and data preview illustrate the virtual table operations:

Read Operation

The data preview of the *users_list_vt_location* virtual table displays the following output:

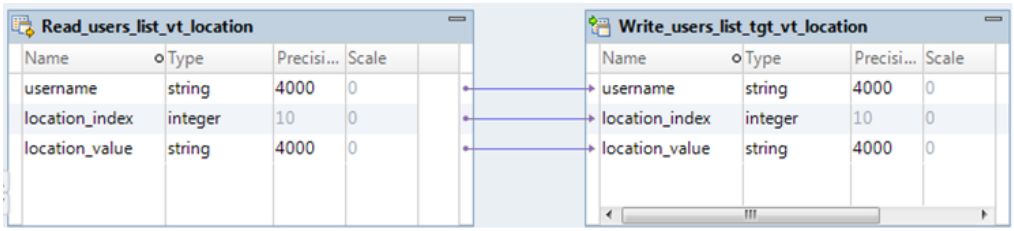
Output

Name: users_list_vt_location

	username	location_index	location_value
1	user2	0	123.88.99.2
2	user2	1	0.0.0.0
3	user4	0	123.123.1.2
4	user4	1	127.0.0.0
5	user1	0	123.123.123.2
6	user1	1	0.0.0.0
7	user3	0	123.123.2.2
8	user3	1	127.0.0.0

Insert Operation

You can select the **Update As Insert** option in the **UpdateMode** advanced property to insert rows into a virtual table. The following pass-through mapping inserts rows into a virtual table:



The mapping source is the *users_list_vt_location* table. The mapping target is the *users_list_tgt_vt_location* table.

The following image displays the data preview of the target virtual table:

Output

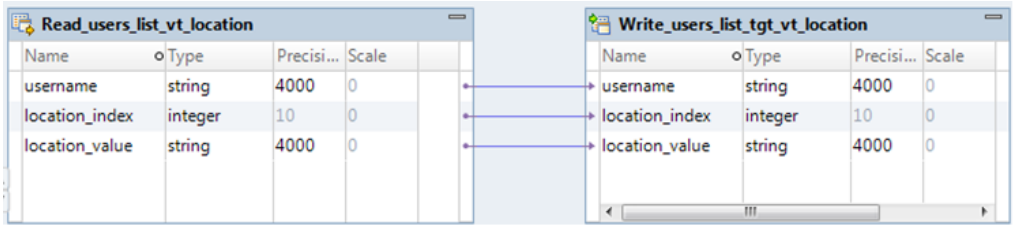
Name: users_list_tgt_vt_location

	username	location_index	location_value
1	user2	0	123.88.99.2
2	user2	1	0.0.0.0
3	user4	0	123.123.1.2
4	user4	1	127.0.0.0
5	user1	0	123.123.123.2
6	user1	1	0.0.0.0
7	user3	0	123.123.2.2
8	user3	1	127.0.0.0

Update Operation

You can select the **Update As Update** option in the **UpdateMode** advanced property to update rows in a virtual table.

The following mapping updates rows in a virtual table:



The mapping source is the *users_list_vt_location* table. The mapping target is the *users_list_tgt_vt_location* table.

CHAPTER 7

Cassandra Run-Time Processing

This chapter includes the following topics:

Cassandra Run-Time Processing Overview

When you create a Cassandra data object read or write operation, you define properties that determine how the Data Integration Service reads data from or writes data to a Cassandra database.

You can configure partitioning and parameterization in the run-time properties.

Filter Expression

To read specific data from a Cassandra table, you can configure a filter condition to query the Cassandra tables. You can use the Native or Platform expression to query specific columns in a Cassandra table.

You can specify a filter expression for columns of the following data types:

- ASCII
- Integer
- Date
- String
- Time
- Timestamp
- TimeUUID
- UUID

Note: If you want to specify filter conditions to query specific columns in a virtual table that represents the renormalized view of a list, you must first run a pass-through mapping with the main table. After you run the

pass-through mapping, you can apply filter conditions to query specific columns in the virtual table for the list.

Native Expression

You can specify a native expression that use AND, OR, or nested conditions. The expression that you enter becomes the WHERE clause in the query used to retrieve records from the source.

An Cassandra filter consists of one or more Boolean expressions. The Boolean expressions uses the following format:

```
<Tablename.columnname><Operator><Value>
```

If you use logical operators, add the operators as a prefix to the expression list. Default is blank.

To filter records from an Cassandra source, set the native expression in the data object read operation.

Note: When you configure a Native expression and select a column that you want to filter, you must delete the `<Tablename.>` value in the native expression.

Platform Expression

You can use the platform expression to select specific records from a Cassandra table based on the filter condition you specify.

The following table describes the properties you specify when you filter records from a Cassandra table when you use the platform expression filter:

Property	Description
Expression Type	The type of filter expression that you want to use to filter records.
Left Field	The Cassandra column on which you want to apply the filter condition.
Operator	Simple operators you can use to filter records. You can select one of the following operators: `=, !=, <, <=, >, and >=`
Right Field	The value you specify to filter Cassandra columns.

Partitioning

When you read data from or write data to Cassandra, you can configure partitioning to optimize the mapping performance at run time. You can configure partitioning for Cassandra mappings that you run in the Spark engine. The partition type controls how the Data Integration Service distributes data among partitions at partition points.

You can define the partition type as key range partitioning. To configure key range partitioning, open the Cassandra data object read operation, and select the **Key Range** partition type option on the **Run-time** tab.

When you configure key range partitioning, the Data Integration Service distributes rows of data based on a port or set of ports that you define as the partition key. You can define a range of values for each port. The Data Integration Service uses the key and ranges to send rows to the appropriate partition.

You can configure a partition key for fields of the following data types:

- Int
- Varint

Parameterization for Cassandra Sources

You can parameterize the Cassandra connection and data object read operation properties to override the mapping properties at run time.

You can parameterize the following read operation properties for a Cassandra source:

- preSQL
- postSQL

Parameterization for Cassandra Targets

You can parameterize the Cassandra connection and data object write operation properties to override the mapping properties at run time.

You can parameterize the following write operation properties for a Cassandra target:

- preSQL
- postSQL

APPENDIX A

Cassandra Data Type Reference

This appendix includes the following topics:

Data Type Reference Overview

Developer Tool uses the following data types in Cassandra mappings:

- Cassandra native data types. Cassandra data types appear in Cassandra definitions in a mapping.
- Transformation data types. Set of data types that appear in the transformations. They are internal data types based on ANSI SQL-92 generic data types, which the Data Integration Service uses to move data across platforms. They appear in all transformations in a mapping.

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

Cassandra and Transformation Data Types

The following table lists the Cassandra data types that Developer Tool supports and the corresponding transformation data types:

Cassandra Data Type	Transformation Data Type	Range and Description for the Transformation Data Type
ASCII	String	1 to 104,857,600 characters.
BIGINT	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
BOOLEAN	String	1 to 104,857,600 characters.

Cassandra Data Type	Transformation Data Type	Range and Description for the Transformation Data Type
DECIMAL	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Cassandra converts decimal values to double in high precision mode.
DATE	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
DOUBLE	Double	Precision 15.
FLOAT	Decimal	For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Secure Agent decimal values to double in high precision mode.
INET	String	1 to 104,857,600 characters.
INT	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10
TEXT	String	1 to 104,857,600 characters.
TIME	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
TIMESTAMP	Date/Time	Jan 1, 0001 A.D. to Dec 31, 9999 A.D. (precision to the nanosecond)
TIMEUUID	String	1 to 104,857,600 characters.
UUID	String	1 to 104,857,600 characters.
VARCHAR	String	1 to 104,857,600 characters.
VARINT	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
LIST	NA	Simba Cassandra JDBC driver re-normalizes the data in the list into virtual tables.
MAP	NA	Simba Cassandra JDBC driver re-normalizes the data in the map into virtual tables.
SET	NA	Simba Cassandra JDBC driver re-normalizes the data in the set into virtual tables.

INDEX