Informatica

Informatica® PowerExchange for Microsoft Azure Data Lake Storage Gen2

10.5.1

User Guide

Informatica PowerExchange for Microsoft Azure Data Lake Storage Gen2 User Guide
10.5.1
September 2021

Publication Date: 2021-10-06

Table of Contents

Preface

Use the *Informatica® PowerExchange® for Microsoft Azure Data Lake Storage Gen2 User Guide* to learn how to read from or write to Microsoft Azure Data Lake Storage Gen2 by using the Developer tool. Learn to create a connection, develop and run mappings and dynamic mappings in the native environment and in the Hadoop and Databricks environments.

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.

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 Microsoft Azure Data Lake Storage Gen2

This chapter includes the following topics:

PowerExchange for Microsoft Azure Data Lake Storage Gen2 Overview

You can use PowerExchange for Microsoft Azure Data Lake Storage Gen2 to connect to Microsoft Azure Data Lake Storage Gen2 from Informatica.

Use PowerExchange for Microsoft Azure Data Lake Storage Gen2 to read data from and write data to Microsoft Azure Data Lake Storage Gen2. You can collate and organize the details from multiple input sources and use PowerExchange for Microsoft Azure Data Lake Storage Gen2 to write data to Microsoft Azure Data Lake Storage Gen2. You can use Microsoft Azure Data Lake Storage Gen2 objects as sources and targets in mappings and dynamic mappings. When you use Microsoft Azure Data Lake Storage Gen2 objects in mappings, you must configure properties specific to Microsoft Azure Data Lake Storage Gen2. You can validate and run mappings in the native and non-native environments.

You can perform read and write operations for secure transfer-enabled storage accounts.

Supported File Formats

The following table describes the file formats that you can use in the native and non-native environments:

File Format	Read	Write	Native	Databricks Spark	Spark
Flat	Yes	Yes	Yes	Yes	Yes
Binary	Yes	Yes	Yes	No	No
Avro (Primitive and hierarchical data types)	Yes	Yes	Yes (Primitive data types only)	Yes	Yes
Json (Primitive and hierarchical data types)	Yes	Yes	Yes (Primitive data types only)	Yes	Yes
ORC (Primitive data types only)	Yes	Yes	Yes	Yes	Yes
Parquet (Primitive and hierarchical data types)	Yes	Yes	Yes (Primitive data types only)	Yes	Yes
Intelligent structure model	Yes	No	No	No	Yes

Introduction to Microsoft Azure Data Lake Storage Gen2

You can use Microsoft Azure Data Lake Storage Gen2 to store data in the form of directories and sub-directories, making it efficient for data access and manipulation.

Microsoft Azure Data Lake Storage Gen2 stores data irrespective of size, structure, and format. Use Microsoft Azure Data Lake Storage Gen2 to process large volumes of data to achieve faster business outcomes. Data scientists and data analysts can use data in the Data Lake to find out specific patterns before you move the analyzed data to a data warehouse. You can use big data analytics available on top of Microsoft Azure Blob Storage.

PowerExchange for Microsoft Azure Data Lake Storage Gen2 Example

You work as a data analyst for a large financial enterprise. The enterprise performs risk management, fraud detection, and other analysis with Azure Data Lake Analytics. You need to write all data toMicrosoft Azure Data Lake Storage Gen2 to perform the analytics.

You can use PowerExchange for Microsoft Azure Data Lake Storage Gen2 and create a mapping to read data from sources such as relational or transactional database or other applications such as Salesforce and write data to Microsoft Azure Data Lake Storage Gen2. After the data is available in the Microsoft Azure Data Lake Storage Gen2, you can perform the data analytics.

CHAPTER 2

PowerExchange for Microsoft Azure Data Lake Storage Gen2 Configuration

This chapter includes the following topics:

PowerExchange for Microsoft Azure Data Lake Storage Gen2 Configuration Overview

PowerExchange for Microsoft Azure Data Lake Storage Gen2 installs with the Informatica services and clients.

To configure PowerExchange for Microsoft Azure Data Lake Storage Gen2, complete the prerequisites.

Prerequisites

Before you use PowerExchange for Microsoft Azure Data Lake Storage Gen2, you must complete the following prerequisites:

- Install and configure the Informatica services.
- Install and configure the Developer tool. You can install the Developer tool when you install Informatica clients.
- Create a Data Integration Service and a Model Repository Service in the Informatica domain.
- Verify that a cluster configuration is created in the domain.
- Verify that a Metadata Access Service is created in the domain.

- Verify that the Hadoop distribution version is 3.x or later.
- Verify that the user used to configure the Informatica domain is added to the cluster and the user has `sudo` privileges when you use non-kerberised Cloudera CDH 6.3 Hadoop distribution.
- Verify that the following tasks are completed before you create a Microsoft Azure Data Lake Storage Gen2 connection:
  - Create an Azure Data Lake Storage Gen2 account and provide **Contributor** or **Reader** role to users.
    The contributor role grants you full access to manage all resources in the storage account, but does not allow you to assign roles.

    The reader role allows you to view all resources in the storage account, but does not allow you to make any changes.

    Note: To add or remove role assignments, you must have write and delete permissions, such as an Owner role.
  - Create an Azure Active Directory application to authenticate users to access the Azure Data Lake Storage Gen2 account. Provide **Storage Blob Data Contributor** or **Storage Blob Data Reader** role to the app.
    The Storage Blob Data Contributor role lets you read, write, and delete Azure Storage containers and blobs in the storage account.

    The Storage Blob Data Reader role lets you only read and list Azure Storage containers and blobs in the storage account.
  - Enable hierarchical namespaces for your Azure Data Lake Storage Gen2 account.
  - Create a file system for Microsoft Azure Data Lake Storage Gen2.
  - To access objects from an HDI 4.0 Kerberised cluster, configure the impersonation user details into your Azure Data Lake Storage Gen2 account. Provide **Contributor** role and `full access`, for the container used in the internal storage account of the HDInsight Data Lake Storage Gen2 cluster, to the impersonation user.

  For more information, see *Azure Data Lake Storage Gen2* documentation.
- To fetch the metadata at design time, you must configure the INFA_PARSER_HOME environment variable for the Metadata Access Service in Informatica Administrator.
  Perform the following steps to configure the INFA_PARSER_HOME property:
  1. Log in to Informatica Administrator.
  2. Click the Metadata Access Service and then click the **Processes** tab on the right pane.
  3. Click **Edit** in the **Environment Variables** section.
  4. Click **New** to add an environment variable.
  5. Enter the name of the environment variable as **INFA_PARSER_HOME**.
  6. Set the value of the environment variable to the absolute path of the Cloudera CDH 6.3 directory on the machine that runs the Metadata Access Service.
     For example:

     INFA_PARSER_HOME = `<Informatica installation directory>/services/shared/hadoop/CDH_6.3`
  7. Recycle the Metadata Access Service.
- To successfully preview data from a local complex file or run a mapping in the native environment, you must configure the INFA_PARSER_HOME property for the Data Integration Service in Informatica Administrator.
  Perform the following steps to configure the INFA_PARSER_HOME property:
  1. Log in to Informatica Administrator.

2. Click the Data Integration Service and then click the **Processes** tab on the right pane.
3. Click **Edit** in the **Environment Variables** section.
4. Click **New** to add an environment variable.
5. Enter the name of the environment variable as **INFA_PARSER_HOME**.
6. Set the value of the environment variable to the absolute path of the Hadoop distribution directory on the machine that runs the Data Integration Service.
7. Recycle the Data Integration Service.

Configure Databricks Connection Advanced Properties

Verify that a Databricks connection is created in the domain. If you want to read NULL values from or write NULL values to an Azure source, configure the following advanced properties in the Databricks connection:

- `infaspark.flatfile.reader.nullValue=True`
- `infaspark.flatfile.writer.nullValue=True`

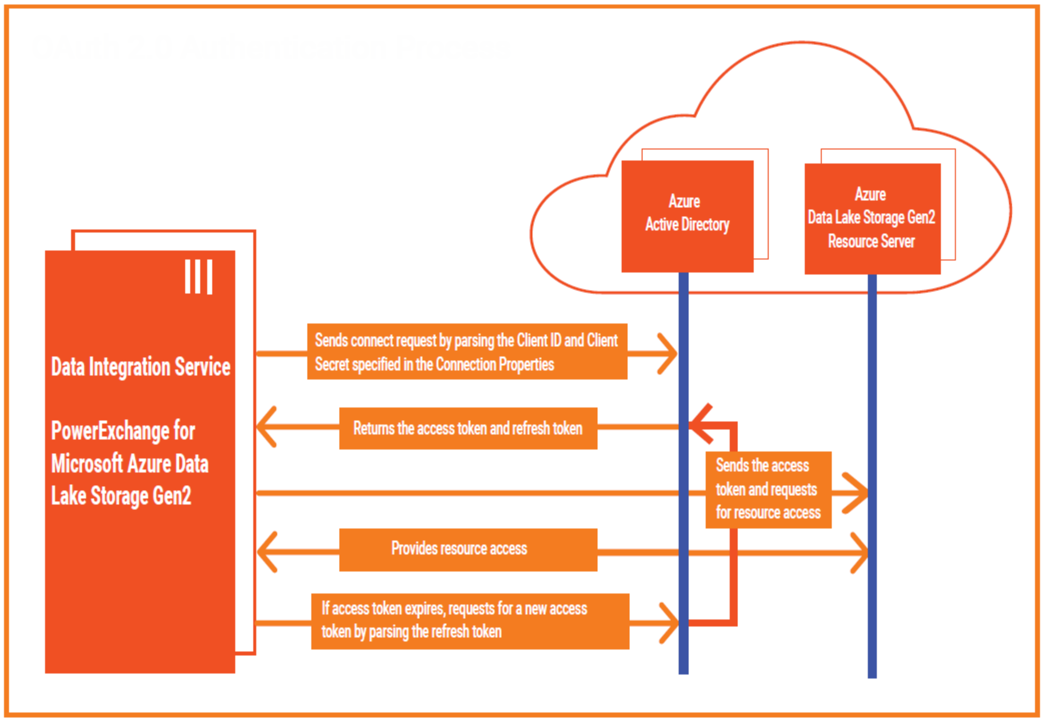
Configure Microsoft Azure Data Lake Storage Gen2 Access in Azure Databricks Cluster

Set the following Hadoop credential configuration options under **Spark Config** in your Databricks cluster configuration to access the Microsoft Azure Data Lake Storage Gen2:

```
spark.hadoop.fs.azure.account.auth.type OAuth
spark.hadoop.fs.azure.account.oauth.provider.type
org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider
spark.hadoop.fs.azure.account.oauth2.client.id <your-service-client-id>
spark.hadoop.fs.azure.account.oauth2.client.secret <your-service-client-secret-key>
spark.hadoop.fs.azure.account.oauth2.client.endpoint https://login.microsoftonline.com/
<directory-ID-of-Azure-AD>/oauth2/token
```

Authentication Process

PoweExchange for Microsoft Azure Data Lake Storage Gen2 uses OAuth 2.0 authorization. The following image shows how does PowerExchange for Azure Data Lake Storage Gen2 receive access tokens and resource access:


Azure Active Directory
Azure Data Lake Storage Gen2 Resource Server
Data Integration Service
PowerExchange for Microsoft Azure Data Lake Storage Gen2
Sends connect request by parsing the Client ID and Client Secret specified in the Connection Properties
Returns the access token and refresh token
Sends the access token and requests for resource access
Provides resource access
If access token expires, requests for a new access token by parsing the refresh token

Installing TLS certificate

If the domain is TLS-enabled, download the certificate first and then add the certificate in the trust store.

Perform the following steps from Developer tool host machine:

1. Run the command to download the certificates and convert them to PEM format.
   - If you use the Azure Government endpoint, run the following command:
     `openssl s_client -servername <ADLS Gen2_instance_name>.dfs.core.usgovcloudapi.net -connect <ADLS Gen2_instance_name>.dfs.core.usgovcloudapi.net:<port number> < /dev/null | openssl x509 -outform pem > gen2.pem`

     Example: `openssl s_client -servername adlsgen2dev.dfs.core.usgovcloudapi.net -connect adlsgen2dev.dfs.core.usgovcloudapi.net:443 < /dev/null | openssl x509 -outform pem > gen2.pem`
   - If you use the Azure non-government endpoint, run the following command:
     `openssl s_client -servername <ADLS Gen2_instance_name>.dfs.core.windows.net -connect <ADLS Gen2_instance_name>.dfs.core.windows.net:<port number> < /dev/null | openssl x509 -outform pem > gen2.pem`

     Example: `openssl s_client -servername bswar.dfs.core.windows.net -connect bswar.dfs.core.windows.net:443 < /dev/null | openssl x509 -outform pem > gen2.pem`
2. Run the following command to import the certificate in the trust store:
   `keytool -import -noprompt -trustcacerts -alias gen2 -file /tmp/test/gen2.pem -keystore <INFA_HOME>/services/shared/security/infa_truststore.jks.`

Configuring the Proxy Server

If your organization uses a proxy server to access the internet, you must configure the proxy server authentication settings for the Data Integration Service. Proxy server settings are applicable to the native environment and on the Spark or Databricks Spark engine.
You can configure both authenticated and unauthenticated proxy servers to use with PowerExchange for Microsoft Azure Data Lake Storage Gen2.

1. Open the Administrator tool.
2. Click the **Administration** tab, and then select the Data Integration Service.
3. Click the **Processes** tab.
4. You can configure the proxy server settings in the advanced properties or custom properties.
   - To configure the proxy server in the advanced properties, add the proxy server properties in the JVM Command Line Options field.
     Add the following properties for HTTPS proxy server:

     ```
     -Dhttps.proxyHost=<Host name of the HTTPS proxy server>

     -Dhttps.proxyPort=<Port number of the HTTPS proxy server>

     -Dhttps.proxyUser=<user name>

     -Dhttps.proxyPassword=<password>

     -Djdk.http.auth.tunneling.disabledSchemes=""
     ```

Add the following properties for HTTP proxy server:

```
-Dhttp.proxyHost=<Host name of the HTTP proxy server>
-Dhttp.proxyPort=<Port number of the HTTP proxy server>
-Dhttp.proxyUser=<user name>
-Dhttp.proxyPassword=<password>
-Djdk.http.auth.tunneling.disabledSchemes=""
```

- To configure the proxy server in the custom properties, add the following properties as JVMOptions:

JVM Option	Property Name	Description
JVMOption1	-Dhttps.proxyHost	Name of the proxy server.
JVMOption2	-Dhttps.proxyPort	Port number of the proxy server. Default is 8080.
JVMOption3	-Dhttps.proxyUser	Required for authenticated proxy. Authenticated user name for the proxy server.
JVMOption4	-Dhttps.proxyPassword	Required for authenticated proxy. Password for the authenticated user.
JVMOption5	-Djdk.http.auth.tunneling.disabledSchemes	Required for authenticated proxy. This property is required to enable the basic authentication.

For example:

```
JVMOption1=-Dhttps.proxyHost=<Host name of the HTTPS proxy server>
JVMOption2=-Dhttps.proxyPort=<Port number of the HTTPS proxy server>
JVMOption3=-Dhttps.proxyUser=<user name>
JVMOption4=-Dhttps.proxyPassword=<password>
JVMOption5=-Djdk.http.auth.tunneling.disabledSchemes=""
```

Java Heap Memory Configuration (Optional)

Configure the memory for the Java heap size in the node that runs the Data Integration Service.

1. In the Administrator tool, navigate to the Data Integration Service for which you want to change the Java heap size.
2. Edit the Custom Properties section in the Data Integration Service Properties.
3. To increase the heap memory size for a large dataset, define the following properties:

   ```
   ExecutionContextOptions.JVMMaxMemory = <size> MB
   ExecutionContextOptions.JVMMinMemory = <size> MB
   ```

   Where <size> is a valid heap size, such as 2048 MB.
4. Click Ok.
5. Restart the Data Integration Service.

CHAPTER 3

Microsoft Azure Data Lake Storage Gen2 Connections

This chapter includes the following topics:

Microsoft Azure Data Lake Storage Gen2 Connection Overview

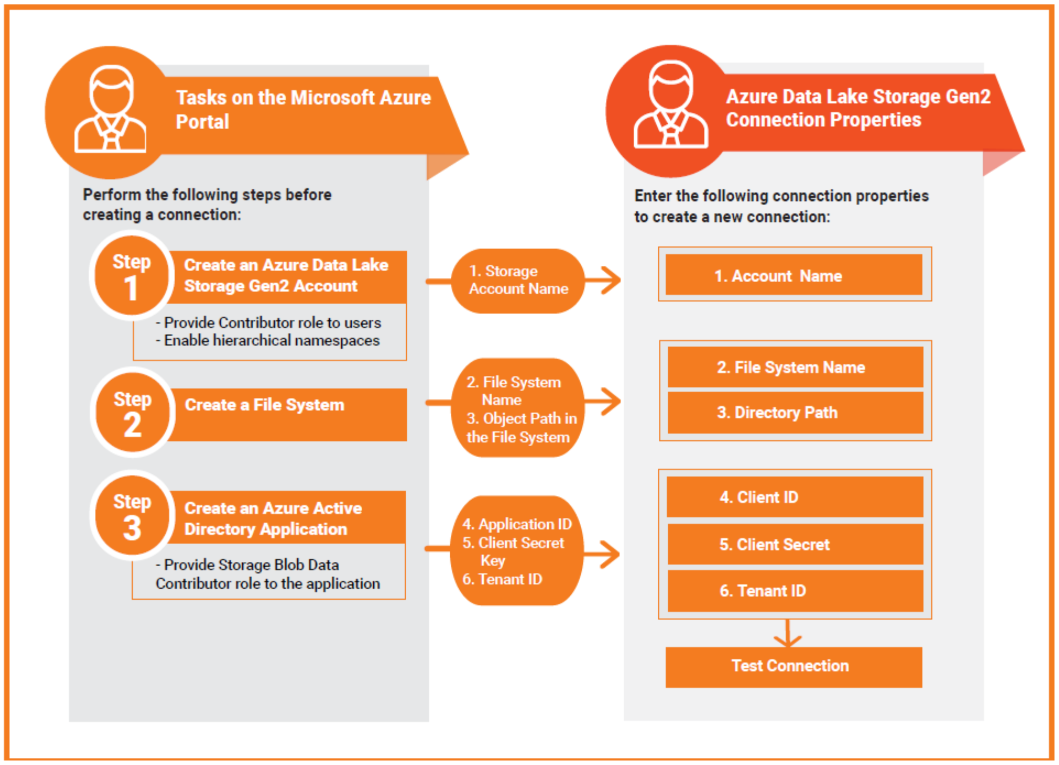
Microsoft Azure Data Lake Storage Gen2 connection enables you to read data from or write data to Microsoft Azure Data Lake Storage Gen2.

You can use Microsoft Azure Data Lake Storage Gen2 connections to create data objects and run mappings. The Developer tool uses the connection when you create a data object. The Data Integration Service uses the connection when you run mappings.

You can create a Microsoft Azure Data Lake Storage Gen2 connection from the Developer tool or the Administrator tool. Create and manage connections in the Preferences dialog box or the Connection Explorer view.

Before you create a connection, ensure that you have completed the prerequisite tasks on the Microsoft Azure portal. The following image shows the mapping between tasks on the Microsoft Azure portal and

Microsoft Azure Data Lake Storage Gen2 connection properties:


Tasks on the Microsoft Azure Portal
Perform the following steps before creating a connection:
Step 1
Create an Azure Data Lake Storage Gen2 Account
- Provide Contributor role to users
- Enable hierarchical namespaces
Step 2
Create a File System
Step 3
Create an Azure Active Directory Application
- Provide Storage Blob Data Contributor role to the application
1. Storage Account Name
2. File System Name
3. Object Path in the File System
4. Application ID
5. Client Secret Key
6. Tenant ID
Azure Data Lake Storage Gen2 Connection Properties
Enter the following connection properties to create a new connection:
1. Account Name
2. File System Name
3. Directory Path
4. Client ID
5. Client Secret
6. Tenant ID
Test Connection

Microsoft Azure Data Lake Storage Gen2 Connection Properties

Use a Microsoft Azure Data Lake Storage Gen2 connection to access a Microsoft Azure Data Lake Storage Gen2.

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

You can create and manage a Microsoft Azure Data Lake Storage Gen2 connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure Data Lake Storage Gen2 connection properties:

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

Property	Description
Location	The domain where you want to create the connection.
Type	The connection type. Select Microsoft Azure Data Lake Storage Gen2.

The following table describes the properties for metadata access:

Property	Description
Account Name	The Microsoft Azure Data Lake Storage Gen2 account name or the service name.
Client ID	The ID of your application to complete the OAuth Authentication in the Azure Active Directory (AD).
Client Secret	The client secret key to complete the OAuth Authentication in the Azure AD.
Tenant ID	The Directory ID of the Azure AD.
File System Name	The name of an existing file system in the Microsoft Azure Data Lake Storage Gen2.
Directory Path	The path of an existing directory without the file system name. There is no default directory. You can select one of the following syntax: - `/` for root directory. - `/dir1` - `dir1/dir2`
Adls Gen2 End-point	Type of Microsoft Azure endpoints. You can select any of the following endpoints: - `core.windows.net`: Default - `core.usgovcloudapi.net`: To select the Azure Government endpoints

For more information about creating a client ID, client secret, tenant ID, and file system name, contact the Azure administrator or see Microsoft Azure Data Lake Storage Gen2 documentation.

Creating a Microsoft Azure Data Lake Storage Gen2 Connection

Create a Microsoft Azure Data Lake Storage Gen2 connection before you create a Microsoft Azure Data Lake Storage Gen2 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 **File System** > **Microsoft Azure Data Lake Storage Gen2**, and click **Add**.
5. Enter a connection name and an optional description.
6. Select Microsoft Azure Data Lake Storage Gen2 as the connection type.
7. Click **Next**.

8. Configure the connection properties.
9. Click **Test Connection** to verify the connection to Microsoft Azure Data Lake Storage Gen2.
10. Click **Finish**.

CHAPTER 4

PowerExchange for Microsoft Azure Data Lake Storage Gen2 Data Objects

This chapter includes the following topics:

Microsoft Azure Data Lake Storage Gen2 Data Object Overview

A Microsoft Azure Data Lake Storage Gen2 data object is a physical data object that represents Microsoft Azure Data Lake Storage Gen2 file as a source or target. A Microsoft Azure Data Lake Storage Gen2 data object is the representation of data that is based on a Microsoft Azure Data Lake Storage Gen2 object. You can configure the data object read and write operation properties that determine how data can be read from Microsoft Azure Data Lake Storage Gen2 or loaded to Microsoft Azure Data Lake Storage Gen2.

To read data from the Microsoft Azure Data Lake Storage Gen2, create a data object read operation based on the Microsoft Azure Data Lake Storage Gen2 data object. Configure the read operation properties to determine how the Data Integration Service must read data from the Microsoft Azure Data Lake Storage Gen2 file. Add the read operation as a Read transformation in a mapping.

To write data to the Microsoft Azure Data Lake Storage Gen2, create a data object write operation based on the Microsoft Azure Data Lake Storage Gen2 data object. Configure the write operation properties to determine how the Data Integration Service must write data to the Microsoft Azure Data Lake Storage Gen2. Add the write operation as a Write transformation in a mapping.

FileName Port

A FileName port is a string port with a default precision of 1024 characters that contains the source path of a file.

You cannot configure the FileName port. You can delete the FileName port if you do not want to read or write the data in the FileName. You cannot create a folder name which contains more than 255 characters.

When you create a data object read or write operation, the FileName port is displayed by default. The following table lists the file sources and the FileName port support for various environments:

File Source	Native Environment	Spark Environment	Databricks Spark Environment
Flat	Yes	Yes	Yes
Binary	Yes	No	No
Avro	Yes	Yes	Yes
JSON	No	No	No
ORC	Yes	Yes	Yes
Parquet	Yes	Yes	Yes

The following table lists the file targets and the FileName port support for various environments:

File Target	Native Environment	Spark Environment	Databricks Spark Environment
Flat	Yes	No	No
Binary	No	No	No
Avro	Yes	Yes	Yes
JSON	No	No	No
ORC	Yes	Yes	Yes
Parquet	Yes	Yes	Yes

Note: You can use the FileName port when you apply Informatica 10.4.0.1 service pack.

Working with FileName Port

You can use the FileName port in a target to dynamically redirect the rows based on the value received by FileName port.

When you run a mapping in the native environment to read or write a flat file using the FileName port, the Data Integration Service creates separate directories for each value in the FileName port in the following format:

```
<CFW FileName>/<CFW FileName>=<valueFromMappingFlow>
```

When you run a mapping that uses the FileName port in the native environment or on the Spark and Databricks Spark engines to read or write a complex file, the Data Integration Service creates separate directories for each value in the FileName port and adds the files within the directories.

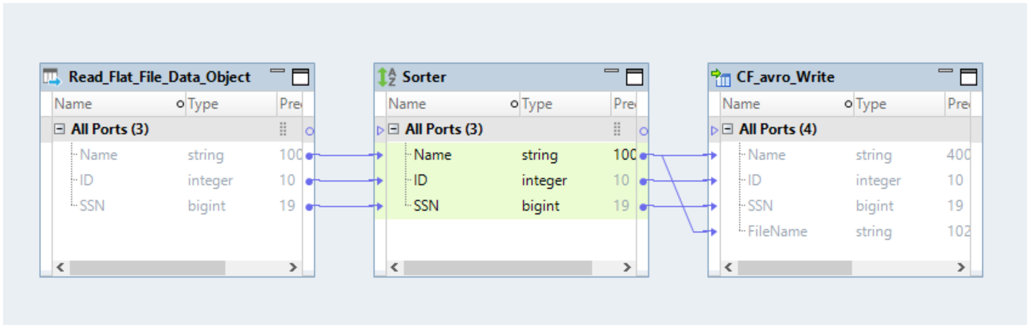
Rules and Guidelines for Using FileName Port

Use the following rules and guidelines when you use the FileName data in the FileName port:

- To read and write complex files, do not use a colon (:) and forward slash (/) character in the file name data of the FileName port of the source or target object.
- To read and write complex files, do not connect FileName port to a FileName port because the FileName port in the source might contain colon (:) and forward slash (/) characters.
- When you map the FileName port to an ID field and the ID value contains NULL, the Data Integration Service creates target files with different names in the native and non-native environments.
  - In the native environment, the target file name is appended with `_EMPTY_`. For example, `target1.avro=_EMPTY_`
  - In the non-native environments, the target file name is appended with `_HIVE_DEFAULT_PARTITION_`. For example, `target1.avro=_HIVE_DEFAULT_PARTITION_`
- If you create a complex file target in the root directory, map the FileName port to an ID field, and run the mapping in the native environment, the Data Integration Service creates a NULL folder in the root directory and places the target file under the NULL folder.
- When you create a mapping to read a flat file, the data preview for the FileName port shows different paths in the native and non-native environments. In the non-native environments, the path also includes ABFSS endpoint details.
- In the Native environment, use the Sorter transformation to sort the source port that you want to map to the FileName port of the Target transformation. After you sort the source port, map the port of the Sorter transformation to the FileName port of the Target transformation. The Data Integration Service creates only one file for each value with the same name. If you do not use the Sorter transformation, the Data Integration Service creates multiple files for each value with the same name.

  For example, create a mapping in the native environment or on the Spark engine to read or write an Avro file using the FileName port.

  The following image shows the Sorter transformation mapping:



Read_Flat_File_Data_Object

Name	Type	Pre
All Ports (3)		
Name	string	100
ID	integer	10
SSN	bigint	19

Sorter

Name	Type	Pre
All Ports (3)		
Name	string	100
ID	integer	10
SSN	bigint	19

CF_avro_Write

Name	Type	Pre
All Ports (4)		
Name	string	400
ID	integer	10
SSN	bigint	19
FileName	string	102

If you want to map the following source port name to the FileName port of the Target transformation and write the data to an Avro target file `target1`:

Name	ID	SSN
Anna	1	1
John	4	4
Smith	4	4
John	5	5
Anna	2	2

Add a Sorter transformation to sort the source port and map the source port to the port of the Sorter transformation. Then, map the port of the Sorter transformation to the FileName port of the Target transformation. The Data Integration Service creates the following directories and single file per thread within the directories:

`target1.avro=Anna`

In this directory, the Data Integration Service creates a file with the following values: `1,1,1,2,2,2`.

`target1.avro=John`

In this directory, the Data Integration Service creates a file with the following values: `4,4,4,5,5,5`.

`target1.avro=Smith`

In this directory, the Data Integration Service creates a file with the following values: `4,4,4`.

If you do not add a Sorter transformation, the Data Integration Service creates the following directories and multiple files within the directories:

`target1.avro=Anna`

In this directory, the Data Integration Service creates two part files with the following values: `1,1,1` and `2,2,2`.

`target1.avro=John`

In this directory, the Data Integration Service creates two files with the following values: `4,4,4` and `5,5,5`.

`target1.avro=Smith`

In this directory, the Data Integration Service creates one file with the following values: `4,4,4`.

Microsoft Azure Data Lake Storage Gen2 Data Object Properties

The Microsoft Azure Data Lake Storage Gen2 Overview view displays general information about the Microsoft Azure Data Lake Storage Gen2 data object and the object properties that apply to the Microsoft Azure Data Lake Storage Gen2 file you import.

General Properties

You can configure the following properties for a Microsoft Azure Data Lake Storage Gen2 data object:

- Name. Name of the Microsoft Azure Data Lake Storage Gen2 data object.

- Description. Description of the Microsoft Azure Data Lake Storage Gen2 data object.
- Connection. Name of the Microsoft Azure Data Lake Storage Gen2 connection.

Microsoft Azure Data Lake Storage Gen2 Data Object Read Operation

The Data Integration Service reads data from a Microsoft Azure Data Lake Storage Gen2 files based on the data object read operation properties that you specify.

When you create a data object read operation, the Developer tool creates a Source transformation and an Output transformation.

The Source transformation represents the data that the Data Integration Service reads from the Microsoft Azure Data Lake Storage Gen2.

The Output transformation represents the data that the Data Integration Service passes into the mapping pipeline.

Directory Source in Microsoft Azure Data Lake Storage Gen2 Sources

You can select the type of source from which you want to read data.

You can select the following type of sources from the **Source Type** option under the advanced properties for a Microsoft Azure Data Lake Storage Gen2 data object read operation:

- File
- Directory

Use the following rules and guidelines to select **Directory** as the source type:

- All the source files in the directory must contain the same metadata.
- The directory read is not applicable to the binary file type.
- All the files must have data in the same format. For example, delimiters, header fields, and escape characters must be same.
- All the files under a specified directory are parsed. The files under subdirectories are not parsed.
- The connector does not perform any validation if there are multiple file formats in the directory you select and might result into errors.

Wildcard Characters

When you run a mapping to read data from an Avro, flat, JSON, ORC, or Parquet file, you can use wildcard characters to specify the source file name.

To use wildcard characters for the source file name, select the **Allow Wildcard Characters** option in the advanced read properties of the Microsoft Azure Data Lake Storage Gen2 data object.

When you run a mapping in the native environment to read a flat file, you can use the `*` wildcard character for the source file name.

When you run a mapping in the native environment or on the Spark and Databricks Spark engine to read an Avro, JSON, ORC, or Parquet file, you can use the `?` and `*` wildcard characters for the source file name.

The question mark character (`?`) allows one occurrence of any character. The asterisk character (`*`) allows zero or more than one occurrence of any character.

Note: The wildcard characters functionality for Microsoft Azure Data Lake Storage Gen2 is available for technical preview. Technical preview functionality is supported for evaluation purposes but is unwarranted and is not production-ready. Informatica recommends that you use in non-production environments only. Informatica intends to include the preview functionality in an upcoming release for production use, but might choose not to in accordance with changing market or technical circumstances. For more information, contact Informatica Global Customer Support.

Wildcard Characters for Reading Data from Flat Files

When you run a mapping in the native environment to read data from a flat file, you can use an asterisk (*) wildcard character to specify the source file name.

You can use the asterisk (*) wildcard to fetch all the files or only the files that match the name pattern. Specify the wildcard character in the following format:

```
abc*.txt
abc.*
```

For example, if you specify `abc*.txt`, all the file names starting with the term `abc` and ending with the `.txt` file extension are read. If you specify `abc.*`, all the file names starting with the term `abc` are read regardless of the extension.

You cannot use the wildcard characters to specify folder names. For example,

```
{ "WildcardURIs": [ "multiread_wildcard/dir1*/", "multiread_wildcard/*/" ] }
```

Wildcard Characters for Reading Data from Complex Files

When you run a mapping in the native environment or on the Spark engine to read data from an Avro, JSON, ORC, or Parquet file, you can use an asterisk (?) and (*) wildcard characters to specify the source file name.

You can use the following wildcard characters:

? (Question mark)

The question mark character (?) allows one occurrence of any character. For example, if you enter the source file name as `a?b.txt`, the Data Integration Service reads data from files with the following names:

- `a1b.txt`
- `a2b.txt`
- `aab.txt`
- `acb.txt`

*** (Asterisk)**

The asterisk mark character (*) allows zero or more than one occurrence of any character. If you enter the source file name as `a*b.txt`, the Data Integration Service reads data from files with the following names:

- `aab.txt`
- `a1b.txt`
- `ab.txt`

- `abc11b.txt`

Note: When you read data from the Avro, JSON, ORC, or Parquet file that contains a colon (:) character in the file name, the mapping fails.

Reading Files from Subdirectories

You can read objects stored in subdirectories in Microsoft Azure Data Lake Storage Gen2 in the native environment or on the Spark engine.

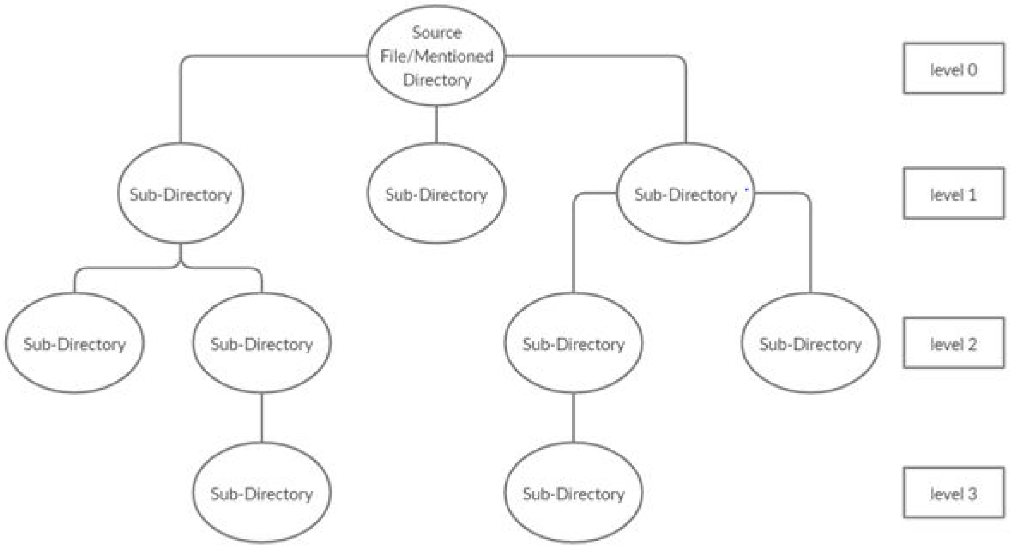
You can use recursive read for flat files in the native environment and on the Spark engine and for complex files only in the native environment.

To enable recursive read, select the source type as **Directory** in the advanced read properties. Configure the **Recursive Directory Depth** advanced read property to specify the number of levels you want to read in the directory. The recursive directory depth must be an integer value.

The following are the possible values for recursive directory depth:

- `Depth = 0`. The Data Integration Service does not read data from any sub-directory.
- `Depth = n`, where `n > 0` and `n < Maximum integer value`.
  The Data Integration Service reads data up to `n` levels of the directory.
- `Depth < 0`. The Data Integration Service reads data from all the available sub-directories.

For example, if the value of recursive directory depth is set to 2, the Data Integration Service will read all the data in the sub-directories available till level 2.



Note: The recursive read functionality for Microsoft Azure Data Lake Storage Gen2 is available for technical preview. Technical preview functionality is supported for evaluation purposes but is unwarranted and is not production-ready. Informatica recommends that you use in non-production environments only. Informatica intends to include the preview functionality in an upcoming release for production use, but might choose not to in accordance with changing market or technical circumstances. For more information, contact Informatica Global Customer Support.

Microsoft Azure Data Lake Storage Gen2 Object Read Operation Properties

The Data Integration Service reads data from a Microsoft Azure Data Lake Storage Gen2 object based on the data object read operation. The Developer tool displays the data object read operation properties of the Microsoft Azure Data Lake Storage Gen2 data object in the Data Object Operation view.

You can view or configure the data object read operation from the source and output properties.

Source properties

Represents data that the Data Integration Service reads from the Microsoft Azure Data Lake Storage Gen2 object. Select the source properties to view data, such as the name and description of the Microsoft Azure Data Lake Storage Gen2 object, the ports, and advanced properties.

Output properties

Represents data that the Data Integration Service passes into the mapping pipeline. Select the output properties to edit the port properties of the data object read operation. You can also use the Tracing Level advanced property to set the amount of detail that the Data Integration Service writes in the log.

Source Properties of the Data Object Read Operation

When you create a data object, the source properties populate based on the Microsoft Azure Data Lake Storage Gen2 object that you add. The source properties of the data object read operation include general, column, and advanced properties that apply to theMicrosoft Azure Data Lake Storage Gen2 object.

You can view the source properties of the data object read operation from the **General**, **Column**, and **Advanced** tabs.

General Properties

The following table describes the source general properties of the data object read operation:

Property	Description
Name	Name of the Microsoft Azure Data Lake Storage Gen2 source object.
Description	Description of the data object read operation.

Ports Properties

The column properties display the data types, precision, and scale of the source property in the data object read operation.

The following table describes the source column properties of the data object read operation:

Property	Description
Name	Name of the column.
Type	Native data type of the column.

Property	Description
Precision	Maximum number of significant digits for Numeric data types, or maximum number of characters for String data types. For numeric data types, precision includes scale.
Scale	Maximum number of digits after the decimal point for numeric values.
Description	Description of the column.

Advanced Properties

You can use the advanced properties to specify data object read operation properties to read data from a Microsoft Azure Data Lake Storage Gen2 server.

The following table describes the Advanced source column properties of the data object read operation:

Property	Description
Tracing Level	Sets the amount of detail that appears in the log file. You can choose terse, normal, verbose initialization, or verbose data. Default is normal.
Concurrent Threads	Number of concurrent connections to read data from Microsoft Azure Data Lake Storage Gen2. When reading a large file or object, you can spawn multiple threads to process data. Configure **Block Size** to partition a large file into smaller parts. Default is 10.
Filesystem Name Override	Overrides the file system name.
Source Type	Select the type of source from which you want to read data. You can select the following source types: - File - Directory Default is File. The directory read is not applicable to binary files.
Allow Wildcard Characters	Indicates whether you want to use wildcard characters for the source file name when you run a mapping to read data from an Avro, flat, JSON, ORC, or Parquet file. For more information about wildcard characters, see "Wildcard Characters" on page 23.
Directory Override	Overrides the default directory path. You can specify an absolute or a relative directory path. - Absolute path. The Data Integration Service searches this directory path in the specified file system. Example of absolute path: `Dir1/Dir2` - Relative path: The Data Integration Service searches this directory path in the native directory path of the object. Example of relative path: `/Dir1/Dir2`
File Name Override	Overrides the file name.
Recursive Directory Depth	Specify the number of sub-directory levels you want to read in the directory in the native environment or on the Spark engine. For more information about recursive directory depth, see "Reading Files from Subdirectories" on page 25.
Block Size	Partitions a large file or object into smaller parts each of specified block size. When reading a large file, consider partitioning a large file into smaller parts and configure **Concurrent Threads** to spawn required number of threads to process data in parallel.

Property	Description
Timeout Interval	The number of seconds to wait when attempting to connect to the server. A timeout will occur if the connection cannot be established in the specified amount of time. Default is 0.
Compression Formats	Compresses data when you read data Microsoft Azure Data Lake Storage Gen2. You can read the following compressed files: - None. No compression format. - Gzip. Applicable only to flat files. The source file extension must be `.GZ`. - Snappy. Applicable only to Avro, ORC, and Parquet file formats in Microsoft Azure Data Lake Storage Gen2. Default is None. **Note:** If you select a `.GZ` source object and select None in the compression format, though the mapping runs successfully, data is not written to the target object.

Schema Properties

The Developer tool displays schema properties for intelligent structure model, Avro, JSON, and Parquet complex file sources in the **Data Object Operations Details** window.

The following table describes the schema properties that you configure for the complex file sources:

Property	Description
Column Name	Displays the name of the column.
Column Type	Displays the format of the column.
Enable Column Projection	Displays the column details of the complex files sources.
Schema Format	Displays the schema format that you selected while creating the complex file data object. You can change the schema format. You can also parameterize the schema format. You can select one of the following options: - Flat - Avro - JSON - ORC - Parquet - Intelligent Structure Model When you read an empty JSON file, the Data Integration Service does not generate headers in the target data. You can change the complex file format without losing the column metadata even after you configure the schema properties for another complex file format. **Note:** You can switch from one schema format to another only once. If you change the schema format more than once, you might lose the original datatypes.

Property	Description
Schema	Displays the schema associated with the complex file. You can select a different schema. You can parameterize the schema or the schema path. To parameterize the schema path, obtain the path from the server. To parameterize schema or schema path for flat files, provide the schema in the same format in which the flat file is imported in the Developer Client. When you use Refresh Schema for the source or target in a mapping and also, parameterize the schema, the parameterized schema takes precedence over the refresh schema. **Note:** If you disable the column projection, the schema associated with the complex file is removed. If you want to associate schema again with the complex file, enable the column projection and select schema.
Schema Properties	Applicable only to flat files.
Column Mapping	Displays the mapping between input and output ports. **Note:** If you disable the column projection, the mapping between input and output ports is removed. If you want to map the input and output ports, enable the column projection and click **Select Schema** to associate a schema to the complex file.

Flat File Schema Properties

You can configure format properties for a flat file that is delimited.

The following table describes the file format and column format properties that you configure for a flat file:

Property	Description
Maximum row to preview	Number of rows to show in data preview. Default is 0.
Delimiters	Character used to separate columns of data. Default is comma. If you enter a delimiter that is the same as the escape character or the text qualifier, you might receive unexpected results. You cannot specify a multibyte character as a delimiter.
Text Qualifier	Quote character that defines the boundaries of text strings. Default is double quotes. If you select a quote character, the Developer tool ignores delimiters within pairs of quotes.
Qualifier Mode	Qualifier behavior for the source object. You can select one of the following options: - **Minimal**. Default mode. Applies qualifier to data that have a delimiter value or a special character present in the data. Otherwise, the Data Integration Service does not apply the qualifier. - **All**. Applies qualifier to all data.
Row Delimiter	Character used to separate the rows of data. You must select the default value, `\012 LF (\n)`.
Header Line Number	Line number that you want to use as the header. You can also read a data from a file that does not have a header. To read data from a file with no header, specify the value of the Header Line Number field as 0.
First Data Row	Line number from where you want the Data Integration Service to read data.

Property	Description
Escape Character	Character immediately preceding a column delimiter character embedded in an unquoted string, or immediately preceding the quote character in a quoted string. When you specify an escape character, the Data Integration Service reads the delimiter character as a regular character. Default is backslash (\).
Retain Escape Character in Data	Not applicable.

Note: If you update the flat file format properties during the data object import and want to see the updated format properties in Data Preview, you must parse the flat file again by selecting the **Schema** property.

Parameterize Column Format Properties

You can parameterize the column format properties for flat files in a parameter file. The following table describes property strings that you can use in a parameter file:

Property	Value
maxRowsToPreview	A positive integer value. Default is 0.
delimiter	Specify the octal code for the character. Preface the octal code with a backslash (\). Specify the following values for the given delimiters: - `\011 TAB` for Tab - `;` for semicolon - `,` for comma - `\040 SP` for space Default is comma. You cannot specify a multibyte character as a delimiter.
textQualifier	Specify the following string values: - SINGLE_QUOTES for `'` - DOUBLE_QUOTES for `"` - NO_QUOTES Default is double quotes.
importColumnFromFirstLine	True or false. Default is true.
rowDelimiter	Default value, `\012 LF (\n)`.
escapeCharacter	A string value. Default is `\`.

A Sample JSON Parameter File

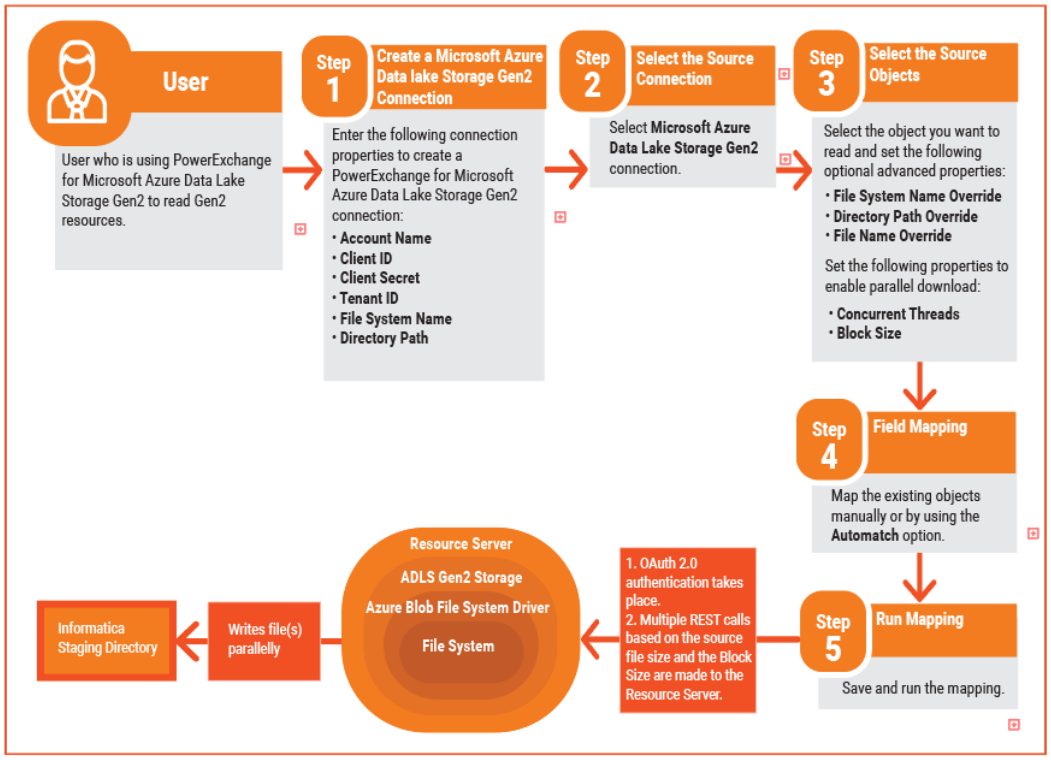
```
{ "maxRowsToPreview": 10, "delimiter": ";", "textQualifier": "SINGLE_QUOTES",
"importColumnFromFirstLine":true, "escapeCharacter" : "-" }
```

Microsoft Azure Data Lake Storage Gen2 Read Use Case

If you want to read large data sets, the task can take a long time to process. You can configure the following read operation properties to partition the source and read the partitions concurrently, which can optimize performance:

- **Block Size**: partitions a large file or object into smaller parts each of specified block size. When reading a large file, consider partitioning a large file into smaller parts and configure **Concurrent Threads** to spawn required number of threads to process data in parallel.
- **Concurrent Threads**: number of concurrent connections to read data from Microsoft Azure Data Lake Storage Gen2. When reading a large file or object, you can spawn multiple threads to process data. Default is 10.
  You must configure **Block Size** if you want multiple threads to process data in parallel.

The following image shows the source properties for parallel read from a large source file:


User
User who is using PowerExchange for Microsoft Azure Data Lake Storage Gen2 to read Gen2 resources.
Step 1
Create a Microsoft Azure Data lake Storage Gen2 Connection
Enter the following connection properties to create a PowerExchange for Microsoft Azure Data Lake Storage Gen2 connection:
• Account Name
• Client ID
• Client Secret
• Tenant ID
• File System Name
• Directory Path
Step 2
Select the Source Connection
Select Microsoft Azure Data Lake Storage Gen2 connection.
Step 3
Select the Source Objects
Select the object you want to read and set the following optional advanced properties:
• File System Name Override
• Directory Path Override
• File Name Override
Set the following properties to enable parallel download:
• Concurrent Threads
• Block Size
Step 4
Field Mapping
Map the existing objects manually or by using the Automatch option.
Step 5
Run Mapping
Save and run the mapping.
1. OAuth 2.0 authentication takes place.
2. Multiple REST calls based on the source file size and the Block Size are made to the Resource Server.
Resource Server
ADLS Gen2 Storage
Azure Blob File System Driver
File System
Writes file(s) parallelly
Informatica Staging Directory

Microsoft Azure Data Lake Storage Gen2 Data Object Write Operation Properties

The Data Integration Service writes data to a Microsoft Azure Data Lake Storage Gen2 object based on the data object write operation. The Developer tool displays the data object write operation properties for the Microsoft Azure Data Lake Storage Gen2 data object in the Data Object Operation section.

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

Input properties

Represent data that the Data Integration Service reads from a Microsoft Azure Data Lake Storage Gen2 directory server. Select the input properties to edit the port properties and specify the advanced properties of the data object write operation.

Target properties

Represent data that the Data Integration Service writes to Microsoft Azure Data Lake Storage Gen2. Select the target properties to view data, such as the name and description of the Microsoft Azure Data Lake Storage Gen2 object.

Input Properties

Input properties represent data that the Data Integration Service writes to a Microsoft Azure Data Lake Storage Gen2 directory server. Select the input properties to edit the port properties of the data object write operation. You can also specify advanced data object write operation properties to write data to Microsoft Azure Data Lake Storage Gen2 objects.

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

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

Ports Properties

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	The name of the port.
Type	The data type of the port.
Precision	The 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.
Detail	The detail of the data type.
Scale	The maximum number of digits after the decimal point for numeric values.
Description	The description of the port.

Run-time Properties

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

The following table describes the run-time properties that you configure for a Microsoft Azure Data Lake Storage Gen2 write operation:

Property	**Description**
Connection	Name of the Microsoft Azure Data Lake Storage Gen2 connection.

Advanced Properties

You can use the advanced properties to specify data object write operation properties to write data to a Microsoft Azure Data Lake Storage Gen2 server.

The following table describes the advanced properties that you configure for a Microsoft Azure Data Lake Storage Gen2 write operation:

Property	**Description**
Tracing Level	By default, the tracing level for every transformation is Normal. Change the tracing level to a Verbose setting when you need to troubleshoot a transformation that is not behaving as expected. Set the tracing level to Terse when you want the minimum amount of detail to appear in the log.
Maintain row order	Not applicable
Concurrent Threads	Number of concurrent connections to write data to Microsoft Azure Data Lake Storage Gen2. When writing a large file or object, you can spawn multiple threads to process data. Configure **Block Size** to partition a large file into smaller parts. Default is 10.
Filesystem Name Override	Overrides the default file system name.
Directory Override	Overrides the default directory path. You can specify an absolute or a relative directory path. - Absolute path. The Data Integration Service searches this directory path in the specified file system. Example of absolute path: `Dir1/Dir2` - Relative path: The Data Integration Service searches this directory path in the native directory path of the object. Example of relative path: `/Dir1/Dir2`
File Name Override	Overrides the file name.
Write Strategy	If the target exists, overwrites or appends the data to the existing file. You can also configure the mapping to fail if the target already exists. **Note: Append** is applicable only to flat files in the native environment.
Block Size	Partitions a large file or object into smaller parts each of specified block size. When writing a large file, consider partitioning the file into smaller parts and configure **Concurrent Threads** to spawn required number of threads to process data in parallel.

Property	Description
Compression Format	Compresses data when you write data to Microsoft Azure Data Lake Storage Gen2. You can compress the data in the following formats: - None. No compression format. - Gzip. Applicable only to flat files. The target file extension must be `.GZ`. - Snappy. Applicable only to Avro, ORC, and Parquet file formats in Microsoft Azure Data Lake Storage Gen2. Default is None.
Timeout Interval	The number of seconds to wait when attempting to connect to the server. A timeout will occur if the connection cannot be established in the specified amount of time. Default is 0.
Stream Rollover Size in GB	Not applicable.
Stream Rollover Time in hours	Not applicable.
Interim Directory	Not applicable.
Partition Option	Select one of the following partition options when you configure a dynamic mapping on the Spark engine: - None. Partitioning is not configured. - Last N Columns Partitioned. The last N columns are selected for partitioning. - Partition Column Names. Comma-separated column names are selected for partitioning.
Partition Arguments	The number or names of partition columns. If you select **None**, as the partition option, do not specify a partition argument. If you select **Last N Columns Partitioned** as the partition option, specify an integer value as the partition argument. If you select **Partition Column Names** as the partition option, specify comma-separated column names as the partition argument.

Schema Properties

The Developer tool displays schema properties for flat files and complex file targets in the **Data Object Operations Details** window.

The following table describes the schema properties that you configure for flat files and complex file targets:

Property	Description
Column Name	Displays the name of the column.
Column Type	Displays the format of the column.
Enable Column Projection	Displays the column details of the complex files sources.

Property	Description
Schema Format	Displays the schema format that you selected while creating the complex file data object. You can change the schema format. You can also parameterize the schema format. You can select one of the following options: - Flat - Avro - JSON - ORC - Parquet You can change the complex file format without losing the column metadata even after you configure the column projection properties for another complex file format. **Note:** You can switch from one schema format to another only once. If you change the schema format more than once, you might lose the original datatypes.
Schema	Displays the schema associated with the complex file. You can select a different schema. You can parameterize the schema or the schema path. To parameterize the schema path, obtain the path from the server. When you use Refresh Schema for the source or target in a mapping and also, parameterize the schema, the parameterized schema takes precedence over the refresh schema. **Note:** If you disable the column projection, the schema associated with the complex file is removed. If you want to associate schema again with the complex file, enable the column projection and select schema.
Schema Properties	Applicable only to flat files.
Column Mapping	Displays the mapping between input and output ports. **Note:** If you disable the column projection, the mapping between input and output ports is removed. If you want to map the input and output ports, enable the column projection and click **Select Schema** to associate a schema to the complex file.

Flat File Schema Properties

You can configure format properties for a flat file that is delimited.

The following table describes the file format and column format properties that you configure for a flat file:

Property	Description
Maximum row to preview	Number of rows to show in data preview. Default is 0.
Delimiters	Character used to separate columns of data. Default is comma. If you enter a delimiter that is the same as the escape character or the text qualifier, you might receive unexpected results. You cannot specify a multibyte character as a delimiter.
Text Qualifier	Quote character that defines the boundaries of text strings. Default is double quotes. If you select a quote character, the Developer tool ignores delimiters within pairs of quotes.

Property	Description
Qualifier Mode	Qualifier behavior for the target object. You can select one of the following options: - **Minimal**. Applies the qualifier to data that contains either a delimiter value or a special character. Otherwise, the Data Integration Service does not apply the qualifier. - **All**. Applies qualifier to all data. Default is minimal.
Row Delimiter	Character used to separate the rows of data. The default value is `\012 LF (\n)`.
Target Header	Indicates whether you want to write data with or without a header.
Escape Character	Character immediately preceding a column delimiter character embedded in an unquoted string, or immediately preceding the quote character in a quoted string. When you specify an escape character, the Data Integration Service reads the delimiter character as a regular character. Default is backslash (\).
Retain Escape Character in Data	Not applicable.

Note: If you update the flat file format properties during the data object import and want to see the updated format properties in Data Preview, you must parse the flat file again by selecting the **Schema** property.

Parameterize Column Format Properties

You can parameterize the column format properties for flat files in a parameter file. The following table describes property strings that you can use in a parameter file:

Property	Value
maxRowsToPreview	A positive integer value. Default is 0.
delimiter	Specify the octal code for the character. Preface the octal code with a backslash (\). Specify the following values for the given delimiters: - `\011 TAB` for Tab - `;` for semicolon - `,` for comma - `\040 SP` for space Default is comma. You cannot specify a multibyte character as a delimiter
textQualifier	Specify the following string values: - SINGLE_QUOTES for ' - DOUBLE_QUOTES for " - NO_QUOTES Default is double quotes.
rowDelimiter	Default value, `\012 LF (\n)`.
escapeCharacter	A string value. Default is `\`.

A Sample JSON Parameter File

```
{ "maxRowsToPreview": 10, "delimiter": ";", "textQualifier": "SINGLE_QUOTES",
"escapeCharacter" : "-" }
```

Rules and guidelines for writing to a flat file target

Consider the following rules when you run a mapping to write data to a flat file in the native environment or on the Spark engine:

- For a mapping that runs on the Spark engine, the first row of the header in the flat file contains an additional `#` symbol. For example, the header `ID_Char` appears as `#ID_Char`.
- For a mapping that runs in the native environment, the header in the flat file does not contain an additional `#` symbol. For example, the header `ID_Char` appears as `ID_Char`.
- For a mapping that runs on the Spark engine, the double quotes text qualifier is honored. For example, the column name `abcd` with the double quotes text qualifier appears as `"abcd"`.
- For a mapping that runs in the native environment, the double quotes text qualifier is not honored. For example, the column name `abcd` with the double quotes text qualifier appears as `abcd`.

Importing a Microsoft Azure Data Lake Storage Gen2 Data Object

Import a Microsoft Azure Data Lake Storage Gen2 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 **Microsoft Azure Data Lake Storage Gen2 Data Object** and click **Next**.

   The **Microsoft Azure Data Lake Storage Gen2 Data Object** dialog box appears.
4. Enter a name for the data object.
5. In the **Resource Format** list, select any of the following formats:
   - Intelligent Structure Model: to read any format that an intelligent structure parses.
   - Binary: to read and write any resource format.
   - Flat: to read and write delimited resources.
   - Avro: to read and write Avro resources.
   - Json: to read and write JSON resources.
   - ORC: to read and write ORC resources.
   - Parquet: to read and write Parquet resources.

   Note: Intelligent structure model is supported only on the Spark engine. For a data object with an intelligent structure model, create a read operation. You cannot use a write transformation for a data object with an intelligent structure model in a mapping. Avro, JSON, ORC, and Parquet are supported in the native and non-native environments.
6. Click **Browse** next to the **Location** option and select the target project or folder.
7. Click **Browse** next to the **Connection** option and select the Microsoft Azure Data Lake Storage Gen2 connection from which you want to import the Microsoft Azure Data Lake Storage Gen2 resource metadata.

8. To add a resource, click **Add** next to the **Selected Resources** option.

   The **Add Resource** dialog box appears.
9. From the Package Explorer, select a naming context from which you want to import the schema.
10. You can Perform one of the following tasks to import an Microsoft Azure Data Lake Storage Gen2 file, and then click **OK**:
    - Navigate to the Microsoft Azure Data Lake Storage Gen2 file that you want to import.
    - Search for the Microsoft Azure Data Lake Storage Gen2 file, enter the name of the Microsoft Azure Data Lake Storage Gen2 file that you want to add and click **OK**.
11. Click **Finish**.
    The data object appears under Data Objects in the project or folder in the **Object Explorer** view.

Creating a Microsoft Azure Data Lake Storage Gen2 Object Read or Write Operation

You can add a Microsoft Azure Data Lake Storage Gen2 data object read or write operation to a mapping or mapplet as a source. You can create the data object read or write operation for one or more Microsoft Azure Data Lake Storage Gen2 data objects.

Before you create a Microsoft Azure Data Lake Storage Gen2 data object read or write operation, you must create at least one Microsoft Azure Data Lake Storage Gen2 data object.

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 read or write operation.
4. Select **Read** or **Write** as the type of data object operation.
5. Click **Add**.

   The **Select Resources** dialog box appears.
6. Select the Microsoft Azure Data Lake Storage Gen2 object for which you want to create the data object read or write operation and click **OK**.
7. Click **Finish**.

The Developer tool creates the data object read or write operation for the selected data object.

Creating a Microsoft Azure Data Lake Storage Gen2 Target

You can create a Microsoft Azure Data Lake Storage Gen2 target using the **Create Target** option.

1. Select a project or folder in the **Object Explorer** view.
2. Select a source or a transformation in the mapping.

3. Right-click the Source transformation and select **Create Target**.
   The **Create Target** dialog box appears.
4. Select **Others** and then select **ADLSGen2** data object from the list in the **Data Object Type** section.
5. Click **OK**.
   The **New ADLSGen2 Data Object** dialog box appears.
6. Enter a name for the data object.
7. In the **Resource Format** list, select any of the following formats to create the target type:
   - Avro
   - Flat
   - JSON
   - ORC
   - Parquet
8. Click **Finish**.

The new target appears under the **Physical Data Objects** category in the project or folder in the **Object Explorer** view.

Rules and Guidelines for Microsoft Azure Data Lake Storage Gen2 Target Data Object

Use the following rules and guidelines when you create a new Microsoft Azure Data Lake Storage Gen2 target:

- You must specify a connection for the newly created Microsoft Azure Data Lake Storage Gen2 target in the **Connection** field to run a mapping.
- When you select a flat resource format that contains different data types and select the **Create Target** option to create a Microsoft Azure Data Lake Storage Gen2 target, the Data Integration Service creates string ports for all the data types with a precision of 256 characters.
- When you select a flat resource format to create a Microsoft Azure Data Lake Storage Gen2 target, the Data Integration Service maps all the data types in the source file to the String data type in the target file. You must manually map the data types in the source and target files.
- For a newly created Microsoft Azure Data Lake Storage Gen2 target, the Data Integration Service considers the value of the root container that you specify in the **Directory** connection property and file name from the **Native Name** property in the Microsoft Azure Data Lake Storage Gen2 data object details. Provide a directory path and file name in the Microsoft Azure Data Lake Storage Gen2 data object read and write advanced properties to overwrite the values.
- When you use a flat resource format to create a target, the Data Integration Service considers the following values for the formatting options:

Formatting Options	Values
Delimiters	Comma (,)
Text Qualifier	No quotes
Import Column Names From First Line	Generates header

Formatting Options	Values
Row Delimiter	Backslash with a character n (\n)
Escape Character	Empty

If you want to configure the formatting options, you must manually edit the projected columns.

CHAPTER 5

Microsoft Azure Data Lake Storage Gen2 Mappings

This chapter includes the following topics:

Microsoft Azure Data Lake Storage Gen2 Mapping Overview

After you create the Microsoft Azure Data Lake Storage Gen2 data object with a Microsoft Azure Data Lake Storage Gen2 connection, you can develop a mapping. You can define the following types of objects in the mapping:

- A Read transformation of the Microsoft Azure Data Lake Storage Gen2 data object to read data from Microsoft Azure Data Lake Storage Gen2.
- A Write transformation of the Microsoft Azure Data Lake Storage Gen2 data object to write data to Microsoft Azure Data Lake Storage Gen2.

Validate and run the mapping. You can deploy the mapping and run it or add the mapping to a Mapping task in a workflow. You cannot define a Lookup transformation of the Microsoft Azure Data Lake Storage Gen2 data object.

Note: If you use multiple connections in a mapping, verify that all connections point to the same Microsoft Azure Data Lake Storage Gen2.

PowerExchange for Microsoft Azure Data Lake Storage Gen2 works with default or internal storage account of the HDInsight Data Lake Storage Gen2 cluster.

When you read data from Microsoft Azure Data Lake Storage Gen2 and terminate the mapping, the Integration Service continues to download the entire data set and then, deletes the data set. This might take a long time based on the size of the data set.

Mapping Validation and Run-time Environments

You can validate and run mappings in the native environment or in a non-native environment, such as Hadoop or Databricks.

When you validate a mapping, you can validate it against one or all of the engines. The Developer tool returns validation messages for each engine.

When you run a mapping, you can choose to run the mapping in the native environment or in a non-native environment, such as Hadoop or Databricks. Configure the run-time environment in the Developer tool to optimize mapping performance and process data that is greater than 10 terabytes. When you run mappings in the native environment, the Data Integration Service processes and runs the mapping. When you run mappings in a non-native environment, the Data Integration Service pushes the processing to a compute cluster, such as Hadoop or Databricks.

You can run standalone mappings, mappings that are a part of a workflow in a non-native environment. When you select the Hadoop environment, the Data Integration Service pushes the mapping logic to the Spark engine.

When you select the Databricks environment, the Integration Service pushes the mapping logic to the Databricks Spark engine, the Apache Spark engine packaged for Databricks.

Directory-Level Partitioning

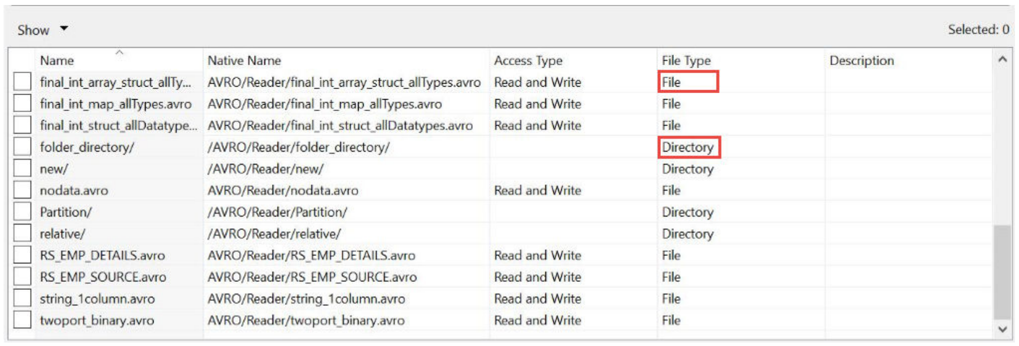
When you run a mapping on the Spark engine, you can read data from and write data to Avro, ORC, and Parquet files that are partitioned based on directories.

You must import a directory that contains only partition folders and select the source type as **Directory** in the advanced read properties.

Importing a data object with partition files

Perform the following steps to import a data object to read or write from partition files:

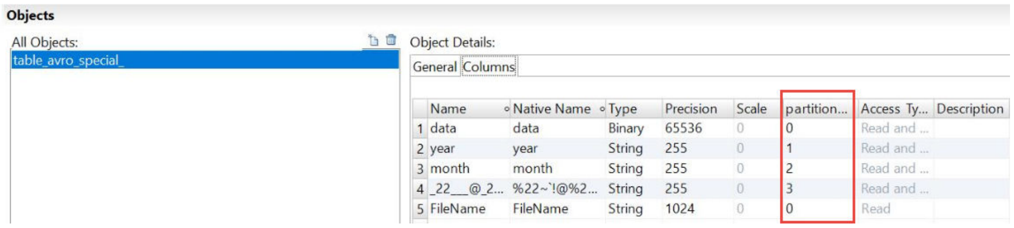
1. Select a project or folder in the **Object Explorer** view.
2. Click **File** > **New** > **Data Object**.
3. Select **Microsoft Azure Data Lake Storage Gen2 Data Object** and click **Next**.
   The **Microsoft Azure Data Lake Storage Gen2 Data Object** dialog box appears.
4. Click **Browse** and select the target project or folder.
5. In the **Resource Format** list, select Avro, Parquet, or ORC.
6. Click **Add** to add a resource to the data object.
   The **Add Resource** dialog box appears. You can use the **File Type** column to distinguish between a directory and a file.



Show ▾ Selected: 0

	Name	Native Name	Access Type	File Type	Description
☐	final_int_array_struct_allTy...	AVRO/Reader/final_int_array_struct_allTypes.avro	Read and Write	File	
☐	final_int_map_allTypes.avro	AVRO/Reader/final_int_map_allTypes.avro	Read and Write	File	
☐	final_int_struct_allDatatype...	AVRO/Reader/final_int_struct_allDatatypes.avro	Read and Write	File	
☐	folder_directory/	/AVRO/Reader/folder_directory/		Directory	
☐	new/	/AVRO/Reader/new/		Directory	
☐	nodata.avro	AVRO/Reader/nodata.avro	Read and Write	File	
☐	Partition/	/AVRO/Reader/Partition/		Directory	
☐	relative/	/AVRO/Reader/relative/		Directory	
☐	RS_EMP_DETAILS.avro	AVRO/Reader/RS_EMP_DETAILS.avro	Read and Write	File	
☐	RS_EMP_SOURCE.avro	AVRO/Reader/RS_EMP_SOURCE.avro	Read and Write	File	
☐	string_1column.avro	AVRO/Reader/string_1column.avro	Read and Write	File	
☐	twoport_binary.avro	AVRO/Reader/twoport_binary.avro	Read and Write	File	

7. Select the check box for a directory. Click **OK**.
8. Click **Finish**.
   The partitioned columns are displayed with the order of partitioning in the data object **Overview** tab.



Objects

All Objects: table_avro_special_

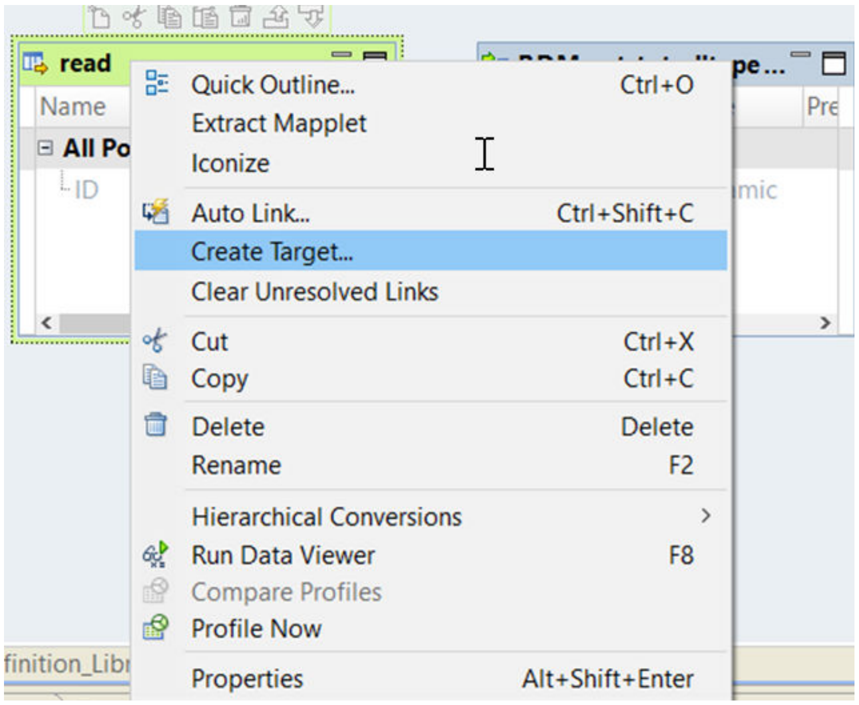
Object Details: General | Columns

	Name	Native Name	Type	Precision	Scale	partition...	Access Ty...	Description
1	data	data	Binary	65536	0	0	Read and ...	
2	year	year	String	255	0	1	Read and ...	
3	month	month	String	255	0	2	Read and ...	
4	_22__@_2...	%22~`!@%2...	String	255	0	3	Read and ...	
5	FileName	FileName	String	1024	0	0	Read	

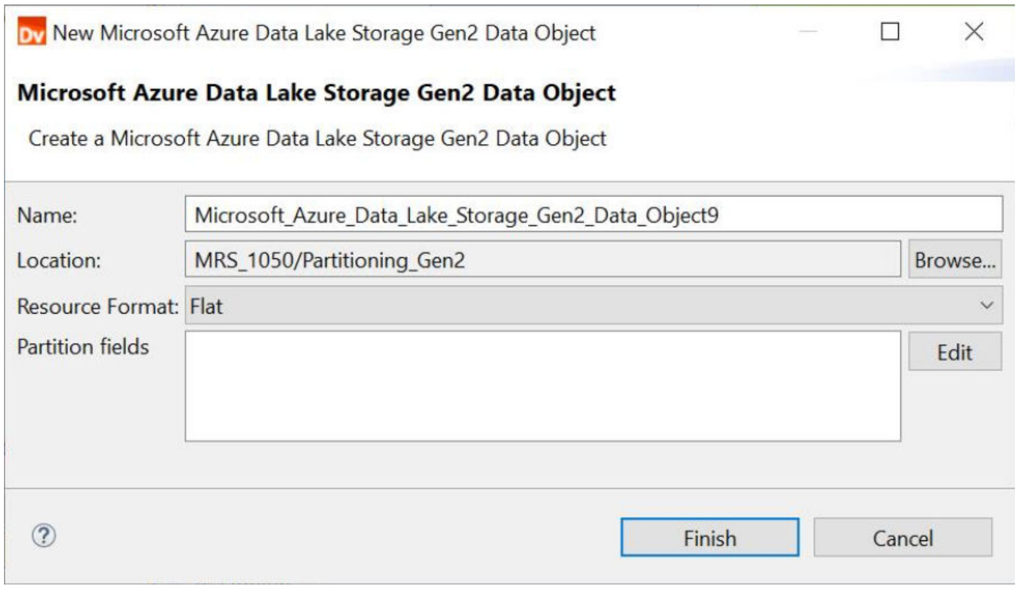
Create target with partition files

Perform the following steps to create target with partition files:

1. Select a project or folder in the **Object Explorer** view.
2. Select a source or a transformation in the mapping.
3. Right-click the Source transformation and select **Create Target**.
   The **Create Target** dialog box appears.



4. Select **Others** and then select **Microsoft Azure Data Lake Storage Gen2** data object from the list in the **Data Object Type** section.
5. Click **OK**.
   The **New Microsoft Azure Data Lake Storage Gen2 Data Object** dialog box appears.



New Microsoft Azure Data Lake Storage Gen2 Data Object

Microsoft Azure Data Lake Storage Gen2 Data Object

Create a Microsoft Azure Data Lake Storage Gen2 Data Object

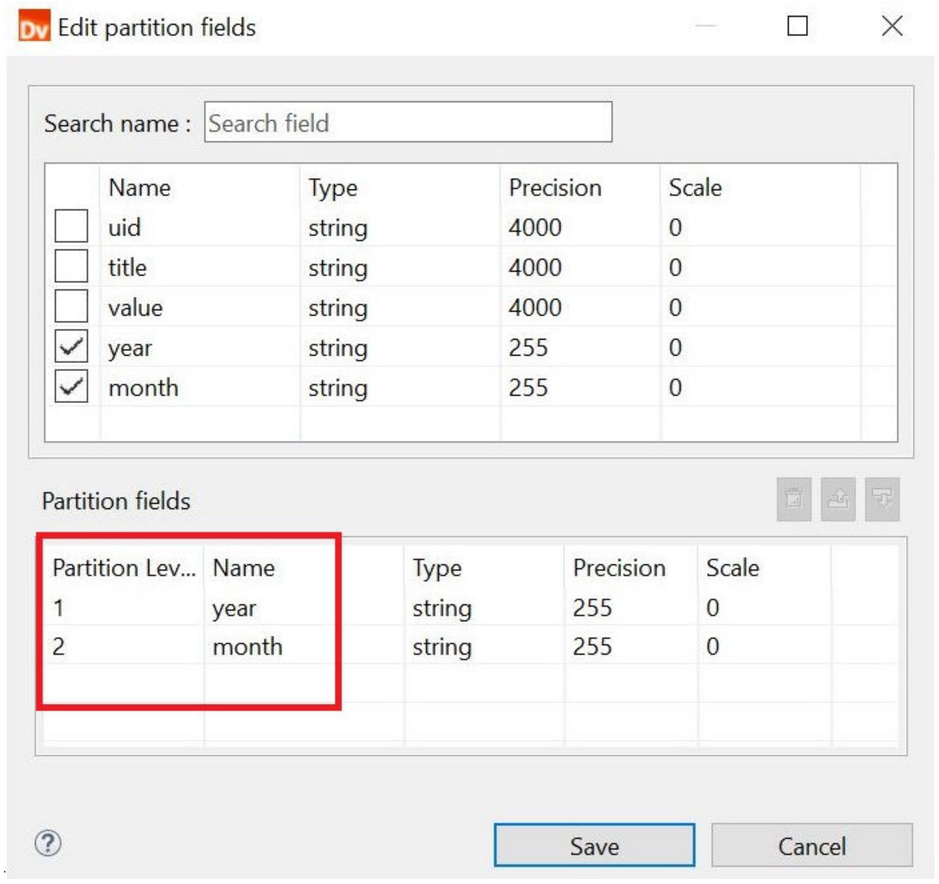
Name: Microsoft_Azure_Data_Lake_Storage_Gen2_Data_Object9

Location: MRS_1050/Partitioning_Gen2

Resource Format: Flat

Partition fields

6. Enter a name for the data object.
7. Enter the partition fields.
   The following image shows the **Edit partition fields** dialog box:



	Name	Type	Precision	Scale
☐	uid	string	4000	0
☐	title	string	4000	0
☐	value	string	4000	0
✓	year	string	255	0
✓	month	string	255	0

Partition Lev...	Name	Type	Precision	Scale
1	year	string	255	0
2	month	string	255	0

8. You can change the partition order using the up and down arrows.
9. Click **Finish**.
   The partitioned columns are displayed with the order of partitioning in the data object **Overview** tab.

Rules and Guidelines for Directory-Level Partitioning

Consider the following rules and guidelines when you read from and write to a partition folder:

- When you import a directory that has a partition folder, the data type for the partition column is imported as a String.
- When you import a Microsoft Azure Data Lake Storage Gen2 object that has partition columns, the partition fields are listed at the end of the list.
- You cannot preview data for partition columns.
- When you create target at the run time, you can add partition fields and arrange the partition columns in an order.
- You cannot use the FileName port when you use directory-level partitioning. Ignore or delete the FileName port.
- When you import a data object, the data and FileName port always show 0 as the partition order.
- The partitioned directory that you select cannot have a partitioned column named FileName. The name is case insensitive.
- If a partition column contains data that has more than 255 characters, the data is truncated and only 255 characters are written in the partition column.

- If a partition column name contains more than 74 characters, the name is truncated and only 74 characters are written in the partition column name.
- The value of the partition directory file path formed using the combination of the partition column name and the target file within the partition directory must not exceed 1024 characters. Otherwise, the mapping will fail.

Microsoft Azure Data Lake Storage Gen2 Mapping Example

You work as a data analyst for a large financial enterprise. The enterprise performs risk management, fraud detection, and other analysis with Microsoft Azure Data Lake Analytics. You need to write the data to Microsoft Azure Data Lake Storage Gen2 to perform the analytics. Create a mapping to read data from a flat file source and write data to Microsoft Azure Data Lake Storage Gen2. After the data is available in the Microsoft Azure Data Lake Storage Gen2, you can perform the data analytics.

You can use the following objects in a Microsoft Azure Data Lake Storage Gen2 mapping:

Flat file input

The input file is a flat file that contains the customer names and other details about customers.

Create a flat file data object. Configure the flat file connection and specify the flat file that contains the customer data as a resource for the data object. Drag the data object into a mapping as a read data object.

Transformations

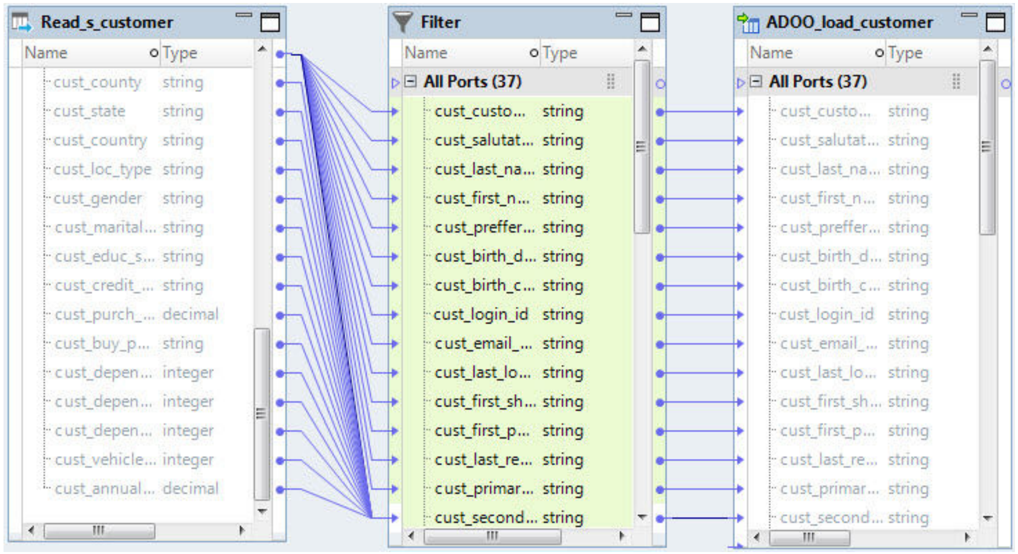
Add Filter transformation to get customer data in a particular region.

The Filter transformation filters the source data based on the value you specify for the region ID column. The Data Integration Service returns the rows that meet the filter condition.

Microsoft Azure Data Lake Storage Gen2 output

Create a Microsoft Azure Data Lake Storage Gen2 data object write operation. Configure the Microsoft Azure Data Lake Storage Gen2 connection and specify the Microsoft Azure Data Lake Storage Gen2 object as a target for the data object. Drag the data object into a mapping as a target data object.

The following image shows the Microsoft Azure Data Lake Storage Gen2 mapping example:



When you run the mapping, the customer records are read from the flat file and written to the Microsoft Azure Data Lake Storage Gen2 file.

Microsoft Azure Data Lake Storage Gen2 Dynamic Mapping Overview

You can use Microsoft Azure Data Lake Storage Gen2 data objects as dynamic sources and targets in a mapping.

Use the Microsoft Azure Data Lake Storage Gen2 dynamic mapping to accommodate changes to source, target, and transformation logics at run time. You can use a Microsoft Azure Data Lake Storage Gen2 dynamic mapping to manage frequent schema or metadata changes or to reuse the mapping logic for data sources with different schemas. Configure rules, parameters, and general transformation properties to create the dynamic mapping.

If the data source for a source or target changes, you can configure a mapping to dynamically get metadata changes at runtime. If a source changes, you can configure the Read transformation to accommodate changes. If a target changes, you can configure the Write transformation accommodate target changes.

You do not need to manually synchronize the data object and update each transformation before you run the mapping again. The Data Integration Service dynamically determine transformation ports, transformation logic in the ports, and the port links within the mapping.

There are the two options available to enable a mapping to run dynamically. You can select one of the following options to enable the dynamic mapping:

- In the **Data Object** tab of the data object read or write operation, select the **At runtime, get data object columns from data source** option when you create a mapping.
  When you enable the dynamic mapping using this option, you can refresh the source and target schemas at the runtime.
- In the **Ports** tab of the data object write operation, select the value of the **Columns defined by** property as **Mapping Flow** when you configure the data object write operation properties.

Note: Dynamic mapping is applicable when you run the mapping in the native environment, on the Spark engine, or on the Databricks Spark engine. When you create a dynamic mapping to read multiple files from a directory and override the directory, verify that the override directory contains a source file with the same name as the imported object. Else, the mapping fails.

For information about dynamic mappings, see the *Informatica Developer Mapping Guide*.

Refresh Schema

You can refresh the source or target schema at the runtime when you enable a mapping to run dynamically. You can refresh the imported metadata before you run the dynamic mapping.

You can enable a mapping to run dynamically using the **At runtime, get data object columns from data source** option in the **Data Object** tab of the Read and Write transformations when you create a mapping.

When you add or override the metadata dynamically, you can include all the existing source and target objects in a single mapping and run the mapping. You do not have to change the source schema to update the data objects and mappings manually to incorporate all the new changes in the mapping.

You can use the mapping template rules to tune the behavior of the execution of such pipeline mapping.

When the Source or Target transformation contains updated ports such as changes in the port names, data types, precision, or scale, the Data Integration Service fetches the updated ports and runs the mapping dynamically. You must ensure that at least one of the column name in the source or target file is the same as before refreshing the schema to run the dynamic mapping successfully.

Even though the original order of the source or target ports in the file changes, the Data Integration Service displays the original order of the ports in the file when you refresh the schemas at runtime.

If there are more columns in the source file as compared to the target file, the Data Integration Service does not map the extra column to the target file and loads null data for all the unmapped columns in the target file.

If the Source transformation contains updated columns that do not match the Target transformation, the Data Integration Service does not link the new ports by default when you refresh the source or target schema. You must create a run-time link between the transformations to link ports at run time based on a parameter or link policy in the **Run-time Linking** tab and update the target schema manually. For information about run-time linking, see the *Informatica Developer Mapping Guide*.

Note: When you refresh a schema of a flat file, the Data Integration Service writes all data types as String data types.

Mapping Flow

You can add all the Source transformation or transformation ports to the target dynamically when enable a mapping to run dynamically using the **Mapping Flow** option. You can then use the dynamic ports in the Write transformation.

When you select the **Mapping Flow** option, the Data Integration Service allows the Target transformation to override ports of the Write transformation with all the updated incoming ports from the pipeline mapping and loads the target file with the ports at runtime.

The Data Integration Service creates the target files dynamically based on the metadata of the incoming ports from the pipeline mapping.

To enable a dynamic mapping using the **Mapping Flow** option, select the value of the **Columns defined by** property as **Mapping Flow** in the **Ports** tab in the Write transformation.

Rules and Guidelines for Microsoft Azure Data Lake Storage Gen2 Dynamic Mappings

Consider the following rules and guidelines for Microsoft Azure Data Lake Storage Gen2 dynamic mappings:

Microsoft Azure Data Lake Storage Gen2 dynamic mappings with directory-level partitioning

- When you parameterize the schema and use run-time linking, ensure that the schema does not have the partitioned data for subsequent mapping runs.
- Ensure that both source and target objects have partitioned files.
- When create target with partition files in Microsoft Azure Data Lake Storage Gen2, the special characters in the column names are written incorrectly to the target. Avoid special characters in column names.

Microsoft Azure Data Lake Storage Gen2 Dynamic Mapping Example

Your organization has a large amount of data that keeps changing. Your organization needs to incorporate all the updated data in a short span of time. Create a dynamic mapping, where you can refresh the source schema dynamically to fetch the updated data. Add all the dynamic ports to the target to override the metadata of the existing ports.

1. Import the Microsoft Azure Data Lake Storage Gen2 read and write data objects.
2. Select a project or folder in the **Object Explorer** view.
3. Click **File** > **New** > **Mapping**.

   The **Mapping** dialog box appears.
4. Enter the name of the mapping in the **Name** field.
5. Click **Finish**.
6. Drag the data object into a mapping.

   The **Microsoft Azure Data Lake Storage Gen2 Data Object Access** dialog box appears.
7. Select the **Read** option and click **OK**.
8. In the **Data Object** tab, select the **At runtime, get data object columns from data source** check box.
9. Drag the data object into a mapping.

   The **Microsoft Azure Data Lake Storage Gen2 Data Object Acess** dialog box appears.
10. Select the **Write** option and click **OK**.
11. In the **Ports** tab, select the value of the **Columns defined by** as **Mapping Flow**.
12. Select all the source incoming ports and add the ports to the target.
13. Save and run the mapping.

APPENDIX A

Microsoft Azure Data Lake Storage Gen2 Datatype Reference

This appendix includes the following topics:

Data Type Reference Overview

Informatica Developer uses the following data types in Microsoft Azure Data Lake Storage Gen2 mappings:

- Microsoft Azure Data Lake Storage Gen2 native data types. Microsoft Azure Data Lake Storage Gen2 data types appear in the physical data object column properties.
- 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. Transformation data types appear in all transformations in a mapping.

When PowerExchange for Microsoft Azure Data Lake Storage Gen2 reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When PowerExchange for Microsoft Azure Data Lake Storage Gen2 writes to a target, it converts the transformation data types to the comparable native data types.

Microsoft Azure Data Lake Storage Gen2 and Transformation Datatypes

The following table lists the Microsoft Azure Data Lake Storage Gen2 data types that the Data Integration Service supports and the corresponding transformation data types:

Microsoft Azure Data Lake Storage Gen2 Native Data Type	Transformation Data Type	Range
Bigint	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision 19, scale 0
Decimal	Decimal	Precision 15
Integer	Integer	-2,147,483,648 to 2,147,483,647 Precision 10, scale 0
String	String	1 to 104,857,600 characters

Flat File and Transformation Data Types

Flat file data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares flat file data types to transformation data types:

Flat File Data type	Transformation Data type	Range
Bigint	Bigint	Precision of 19 digits, scale of 0
Number	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 Data Integration Service converts decimal values to double in high precision mode. If the precision is greater than 15, the Data Integration Service converts decimal values to double in low-precision mode.
String	String	1 to 104,857,600 characters
Nstring	String	1 to 104,857,600 characters

Avro Data Types and Transformation Data Types

Avro data types map to transformation data types that the Data Integration Service uses to move data across platforms.

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

Avro Data Type	Transformation Data Type	Range
Array	Array	Unlimited number of characters.
Boolean	Integer	TRUE (1) or FALSE (0).
Bytes	Binary	Precision 4000.
Date	Date/Time	January 1, 0001 to December 31, 9999.
Decimal	Decimal	Decimal value with declared precision and scale. Scale must be less than or equal to precision. For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Double	Double	Precision 15.
Fixed	Binary	1 to 104,857,600 bytes.
Float	Double	Precision 15.
Int	Integer	-2,147,483,648 to 2,147,483,647 Precision 10 and scale 0.
Long	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807. Precision 19 and scale 0.
Map	Map	Unlimited number of characters.
Record	Struct	Unlimited number of characters.
String	String	1 to 104,857,600 characters.
Time	Date/Time	Time of the day. Precision to microsecond.

Avro Data Type	Transformation Data Type	Range
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
Union	Corresponding data type in a union of ["primitive_type|complex_type", "null"] or ["null", "primitive_type|complex_type"].	Dependent on primitive or complex data type.

Avro Union Data Type

A union indicates that a field might have more than one data type. For example, a union might indicate that a field can be a string or a null. A union is represented as a JSON array containing the data types.
The Developer tool only interprets a union of ["primitive_type|complex_type", "null"] or ["null", "primitive_type|complex_type"]. The Avro data type converts to the corresponding transformation data type.

Avro Timestamp Data Type Support

The following table lists the Timestamp data type support for Avro file formats:

Timestamp Data type	Native	Spark
Timestamp_micros	Yes	Yes
Timestamp_millis	Yes	No
Time_millis	Yes	No
Time_micros	Yes	No

Unsupported Avro Data Types

The Developer tool does not support the following Avro data types:

- Enum
- Null
- Timestamp_tz

JSON Data Types and Transformation Data Types

JSON data types map to transformation data types that the Data Integration Service uses to move data across platforms.

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

JSON	Transformation	Range
Array	Array	Unlimited number of characters.
Double	Double	Precision of 15 digits.
Integer	Integer	-2,147,483,648 to 2,147,483,647. Precision of 10, scale of 0.
Object	Struct	Unlimited number of characters.
String	String	1 to 104,857,600 characters.

Unsupported JSON Data Types

The Developer tool does not support the following JSON data types:

- Date
- Decimal
- Timestamp
- Enum
- Union

ORC Data Types and Transformation Data Types

ORC file data types map to transformation data types that the Data Integration Service uses to move data across platforms.

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

ORC File Data Type	Transformation Data Type	Range and Description
BigInt	BigInt	-9223372036854775808 to 9,223,372,036,854,775,807.
Boolean	Integer	TRUE (1) or FALSE (0).
Char	String	1 to 104,857,600 characters.

ORC File Data Type	Transformation Data Type	Range and Description
Date	Date/Time	January 1, 0001 to December 31, 9999.
Double	Double	Precision of 15 digits.
Float	Double	Precision of 15 digits.
Integer	Integer	-2,147,483,648 to 2,147,483,647.
SmallInt	Integer	-32,768 to 32,767.
String	String	1 to 104,857,600 characters.
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
TinyInt	Integer	-128 to 127.
Varchar	String	1 to 104,857,600 characters.

When you run a mapping on the Spark or Databricks Spark engine to write an ORC file to a target, the Data Integration Service writes the data of the Char and Varchar data types as String.

Note: You can use ORC data types to read and write complex file objects in mappings that run on the Spark engine only.

Unsupported ORC Data Types

The Developer tool does not support the following ORC data types:

- Map
- List
- Struct
- Union

Parquet Data Types and Transformation Data Types

Parquet data types map to transformation data types that the Data Integration Service uses to move data across platforms.

The following table compares the Parquet data types that the Data Integration Service supports and the corresponding transformation data types:

Parquet	Transformation	Range
Binary	Binary	1 to 104,857,600 bytes
Binary (UTF8)	String	1 to 104,857,600 characters
Boolean	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0
Date	Date/Time	January 1, 0001 to December 31, 9999.
Decimal	Decimal	Decimal value with declared precision and scale. Scale must be less than or equal to precision. For transformations that support precision up to 38 digits, the precision is 1 to 38 digits, and the scale is 0 to 38. For transformations that support precision up to 28 digits, the precision is 1 to 28 digits, and the scale is 0 to 28. If you specify the precision greater than the maximum number of digits, the Data Integration Service converts decimal values to double in high precision mode.
Double	Double	Precision of 15 digits.
Float	Double	Precision of 15 digits.
Int32	Integer	-2,147,483,648 to 2,147,483,647 Precision of 10, scale of 0
Int64	Bigint	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 Precision of 19, scale of 0
Map	Map	Unlimited number of characters.
Struct	Struct	Unlimited number of characters.
Time	Date/Time	Time of the day. Precision to microsecond.

Parquet	Transformation	Range
Timestamp	Date/Time	January 1, 0001 00:00:00 to December 31, 9999 23:59:59.997. Precision to microsecond.
group (LIST)	Array	Unlimited number of characters.

The Parquet schema that you specify to read or write a Parquet file must be in smaller case. Parquet does not support case-sensitive schema.

Parquet Timestamp Data Type Support

The following table lists the Timestamp data type support for Parquet file formats:

Timestamp Data type	Native	Spark
Timestamp_micros	Yes	No
Timestamp_millis	Yes	No
Time_millis	Yes	No
Time_micros	Yes	No
int96	Yes	Yes

Unsupported Parquet Data Types

The Developer tool does not support the following Parquet data types:

- Timestamp_nanos
- Time_nanos
- Timestamp_tz

Rules and Guidelines for Data Types

Consider the following rules and guidelines for data types:

- Avro data types support:
  - Date, Decimal, and Timestamp data types are applicable when you run a mapping in the native environment or on the Spark engine in Cloudera CDH 6.3 distribution.
  - Time data type is applicable when you run a mapping in the native environment in Cloudera CDH 6.3 distribution.

- JSON data types support:
  - For PowerExchange for Microsoft Azure Data Lake Storage Gen2, you can read and write complex file objects in JSON format in mappings that run in the native environment, Spark engine, and Databricks Spark engine.
    For other file-based adapters, you can read and write complex file objects in JSON format in mappings that run on the Spark engine only.
- Parquet data types support:
  - Before you create and run a new mapping on the Databricks engine to read a parquet file with hierarchical data types, you must set the `-DINFA_HADOOP_DIST_DIR=hadoop\Databricks_7.2` option in the `developerCore.ini` file.
  - When you set the `-DINFA_HADOOP_DIST_DIR=hadoop\<Distro>` option in the `developerCore.ini` file and import a Parquet file, the format of the imported metadata differs based on the distribution. For Cloudera CDP 7.1, the metadata is imported as string and for other supported distributions, the metadata is imported as UTF8.
  - Date, Time, and Timestamp data types till microseconds are applicable when you run a mapping in the native environment , Blaze, and Spark engine in the Hortonworks HDP 3.1, Azure HDInsight HDI 4.0, and Cloudera CDP 7.1 distributions.
  - Date, Time_Millis, and Timestamp_Millis data types are applicable when you run a mapping in the native environment or Spark engine in MapR 6.1.
  - Decimal data types are applicable when you run a mapping in the native environment and Spark engine in Cloudera CDH 6.3, Hortonworks HDP 3.1, Amazon EMR 5.20, MapR 6.1, and Azure HDInsight HDI 4.0 distributions.
  - Date, Time, Timestamp, and Decimal data types are applicable when you run a mapping on the Databricks Spark engine.
  - When you run a mapping and use Date data type that does not have a time value, the Data Integration Service adds the time value, based on the time zone, to the date in the target.
    For example, Date data type used in the source:
    ```
    1980-01-09
    ```
    Value generated in the target:
    ```
    1980-01-09 00:00:00
    ```
  - When you run a mapping in the native environment and use Time data type in the source, the Data Integration Service writes incorrect date value to the target.
    For example, Time data type used in the source:
    ```
    1980-01-09 06:56:01.365235000
    ```
    Incorrect Date value is generated in the target:
    ```
    1899-12-31 06:56:01.365235000
    ```
  - When you run a mapping in the native environment and use Date data type in the source, the Data Integration Service writes incorrect time value to the target.
    For example, Date data type used in the source:
    ```
    1980-01-09 00:00:00
    ```
    Incorrect Time value generated in the target:
    ```
    1980-01-09 05:30:00
    ```
- To run a mapping that reads and writes Date, Time, Timestamp, and Decimal data types, update the `-DINFA_HADOOP_DIST_DIR` option to the `developerCore.ini` file. The `developerCore.ini` file is located in the following directory:

```
<Client installation directory>\clients\DeveloperClient\
```

Add the following path to the `developerCore.ini` file:

```
-DINFA_HADOOP_DIST_DIR=hadoop\<Hadoop distribution>_<version>
```

For example: `-DINFA_HADOOP_DIST_DIR=hadoop\CDH_6.3`

- To use precision up to 38 digits for Decimal data type in the native environment, set the `EnableSDKDecimal38` custom property to `true` for the Data Integration Service. The `EnableSDKDecimal38` custom property is applicable to all file-based PowerExchange adapters except PowerExchange for HDFS.

INDEX