Informatica

Informatica® PowerExchange for Microsoft Azure SQL Data Warehouse

10.5.6

User Guide

Informatica PowerExchange for Microsoft Azure SQL Data Warehouse User Guide
10.5.6
May 2024

Publication Date: 2024-05-30

Table of Contents

Preface

Use the *Informatica® PowerExchange® for Microsoft Azure SQL Data Warehouse User Guide* to learn how to read from or write to Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse

This chapter includes the following topics:

PowerExchange for Microsoft Azure SQL Data Warehouse Overview

Use PowerExchange for Microsoft Azure SQL Data Warehouse to read data from and write data to Microsoft Azure SQL Data Warehouse or Azure Synapse SQL. You can also use PowerExchange for Microsoft Azure SQL Data Warehouse to collate and organize the details from multiple input sources and write the data to Microsoft Azure SQL Data Warehouse. You can read data from and write data to Microsoft Azure SQL Data Warehouse case-sensitive databases also.

You can use Microsoft Azure SQL Data Warehouse objects as sources and targets in mappings and dynamic mappings. When you use Microsoft Azure SQL Data Warehouse objects in mappings, you must configure properties specific to Microsoft Azure SQL Data Warehouse. You can validate and run mappings in native or Hadoop environments. You can run Microsoft Azure SQL Data Warehouse mappings in a Hadoop environment on Kerberos enabled clusters.

PowerExchange for Microsoft Azure SQL Data Warehouse is optimized for large data sets and can perform better than traditional data integration methods, such as ODBC or JDBC. When you read data from or write data to a Microsoft Azure SQL Data Warehouse, PowerExchange for Microsoft Azure SQL Data Warehouse stages data files to Microsoft Azure Storage and uses T-SQL commands with Microsoft Polybase to load relational and non-relational data in parallel. PowerExchange for Microsoft Azure SQL Data Warehouse supports Azure tables.

PowerExchange for Microsoft Azure SQL Data Warehouse does not support Azure views and synonyms.

Note: You cannot use a private endpoint to connect to Microsoft Azure SQL Data Warehouse.

Example

You work in sales operations and you frequently need to analyze a high volume of data to improve operational intelligence. You design a mapping to read data from Salesforce and other transactional systems and aggregate the data. You create a summary table in Microsoft Azure SQL Data Warehouse that you can query against to assess your sales organization's performance.

You use PowerExchange for Microsoft Azure SQL Data Warehouse to write aggregated data to Microsoft Azure SQL Data Warehouse.

Microsoft Azure SQL Data Warehouse

Azure SQL Data Warehouse is a cloud-based, scalable database that can process massive volumes of relational and non-relational data. Microsoft Azure SQL Data Warehouse is built on a massively parallel processing (MPP) architecture to handle the enterprise workload.

CHAPTER 2

PowerExchange for Microsoft Azure SQL Data Warehouse Configuration

This chapter includes the following topics:

PowerExchange for Microsoft Azure Data Warehouse Configuration Overview

When you configure the PowerExchange for Microsoft Azure SQL Data Warehouse server component, you enable the Data Integration Service to read data from or write data to Microsoft Azure SQL Data Warehouse.

Prerequisites

Before you can use PowerExchange for Microsoft Azure SQL Data Warehouse, verify that either the `db_owner` privilege or the following more granular privileges are granted to the user to connect to the Microsoft Azure SQL Data Warehouse and perform read and write operations successfully:

- `EXEC sp_addrolemember 'db_datareader', '<user>';` // Alternately assign permission to individual table
- `EXEC sp_addrolemember 'db_datawriter', '<user>';` // Alternately assign permission to individual table
- `GRANT ALTER ANY EXTERNAL DATA SOURCE TO <user>;`

- `GRANT ALTER ANY EXTERNAL FILE FORMAT TO <user>;`
- `GRANT CONTROL TO <user>;`
- `GRANT CREATE TABLE TO <user>;`

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 Blob Storage Access in Azure Databricks Cluster

Verify that a cluster configuration is created in the domain. Specify your Microsoft Azure Blob Storage account name and account key under **Spark Config** in your Databricks cluster configuration to access the Microsoft Azure Blob Storage. Add `spark.hadoop` as a prefix to the Hadoop configuration key as shown in the following text:

```
spark.hadoop.fs.azure.account.key.<your-storage-account-name>.blob.core.windows.net
<your-storage-account-access-key>
```

Note: In case of multiple Microsoft Azure Blob Storage accounts, you must configure the account name and account key for each of the Azure Blob Storage account.

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

Verify that a cluster configuration is created in the domain. Specify your Microsoft Azure Data Lake Storage Gen2 account name and account key under **Spark Config** in your Databricks cluster configuration to access the Microsoft Azure Data Lake Storage Gen2. Add `spark.hadoop` as a prefix to the Hadoop configuration key as shown in the following text:

```
spark.hadoop.fs.azure.account.key.<your-storage-account-name>.dfs.core.windows.net <your-
storage-account-access-key>
```

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 following command to download the certificate available at `<AzureDWInstance>.database.windows.net` on the port `1433`:

   ```
   openssl s_client -servername <AzureDWInstance>.database.windows.net -connect
   <AzureDWInstance>.database.windows.net:<port number> < /dev/null | openssl x509 -
   outform pem > azureDW.pem.
   ```

2. Download and install all four certificates available at https://www.microsoft.com/pki/mscorp/cps/default.htm.

3. Run the following commands to import the certificates into the trust store:

```
keytool -import -noprompt -trustcacerts -alias azuredw -file <Certificate path>/
azureDW.pem -keystore <INFA_HOME>/services/shared/security/infa_truststore.jks
```

```
keytool -import -keystore $INFA_HOME/services/shared/security/infa_truststore.jks -
storepass pass2038@infaSSL -alias msft-ittlsca1 -file "<Cerificate path>/Microsoft
IT TLS CA 1.crt"
```

```
keytool -import -keystore $INFA_HOME/services/shared/security/infa_truststore.jks -
storepass pass2038@infaSSL -alias msft-ittlsca2 -file "<Cerificate path>/Microsoft
IT TLS CA 2.crt"
```

```
keytool -import -keystore $INFA_HOME/services/shared/security/infa_truststore.jks -
storepass pass2038@infaSSL -alias msft-ittlsca4 -file "<Cerificate path>/Microsoft
IT TLS CA 4.crt"
```

```
keytool -import -keystore $INFA_HOME/services/shared/security/infa_truststore.jks -
storepass pass2038@infaSSL -alias msft-ittlsca5 -file "<Cerificate path>/Microsoft
IT TLS CA 5.crt"
```

4. Restart the Data Integration Service.

Azure Active Directory Authentication

You can configure Azure Active Directory (AAD) authentication to connect to Microsoft Azure SQL Data Warehouse. Default is Microsoft SQL Server authentication.

To configure the AAD authentication, perform the following tasks:

Import Server Certificate

Applicable if a trust store file is not configured for your organization and you want to use AAD authentication with Active Directory Federation Services in Azure. Import the server certificate to the following location:

Use the following command to import the certificate:

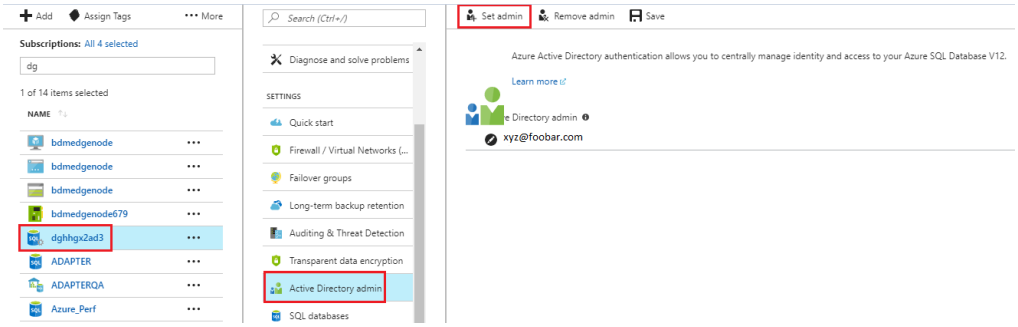
```
keytool -import -trustcacerts -alias <alias name of the certificate> -file <certificate file
path> -keystore -storepass <password for the truststore>
```

Set Admin

Perform the following steps to set admin between Microsoft SQL Server that has the Microsoft Azure SQL Data Warehouse hosted and the Azure Active Directory:

1. Log on to the Microsoft Azure portal using your credentials. The Dashboard page appears.
2. From the All Resources page, select the Microsoft SQL Server that has the Microsoft Azure SQL Data Warehouse hosted.

3. Select the **Active Directory admin** option under Settings displayed for the Microsoft SQL Server. The image shows the Active Directory admin settings:



4. Click **Set admin**. The Add admin page appears.
5. Enter the email ID that you want to use as admin and click **Select**.
6. Click **Save**.

Create a User

Perform the following steps to create a user:

1. Connect to the Microsoft Azure SQL Data Warehouse using the Azure Active Directory admin created in the previous steps. You can use Microsoft SQL Server Management Studio to connect to the Microsoft Azure SQL Data Warehouse.
2. Type and run the following command to create a user: `create user [user@foobar.com] from external provider;`
   For more information, see Microsoft Azure documentation.
3. Assign the required privileges to the user.

Configure the JDBC URL and the user you created in connection properties to enable AAD authentication.

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.

Configure Temporary Directory Location (Optional)

Follow below steps to configure the temporary directory location 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 temporary directory location.
2. Click the **Processes** tab.
3. Click **Custom Properties**. The **Edit Custom Properties** dialog box appears.
4. Click **New** to add a new custom property.
5. Add the JVMOption custom property for the Data Integration Service and specify the value in the following format:

   ```
   -Djava.io.tmpdir=<required tmp directory location>
   ```

   For example,

   ```
   Property Name: JVMOption1
   Value: -Djava.io.tmpdir=/opt/Informatica/tmp/ZUDAP/
   ```
6. Click **Ok**.
7. Restart the Data Integration Service.

CHAPTER 3

Microsoft Azure SQL Data Warehouse Connections

This chapter includes the following topics:

Microsoft Azure SQL Data Warehouse Connections Overview

Microsoft Azure SQL Data Warehouse connection enables you to read data from or write data to Microsoft Azure SQL Data Warehouse.

You can use Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse connection from the Developer tool or the Administrator tool. The Developer tool stores connections in the domain configuration repository. Create and manage connections in the Preferences dialog box or the Connection Explorer view.

Microsoft Azure SQL Data Warehouse Connection Properties

Use a Microsoft Azure SQL Data Warehouse connection to access a Microsoft Azure SQL Data Warehouse.

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 SQL Data Warehouse connection in the Administrator tool or the Developer tool. The following table describes the Microsoft Azure SQL Data Warehouse connection properties:

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

The following table describes the properties for metadata access:

Property	Description
Azure DW JDBC URL	Microsoft Azure Data Warehouse JDBC connection string. For example, you can enter the following connection string: `jdbc:sqlserver:// <Server>.database.windows.net:1433;database=<Database>` The Administrator can download the URL from Microsoft Azure portal.
Azure DW JDBC Username	User name to connect to the Microsoft Azure SQL Data Warehouse account. You must have permission to read, write, and truncate data in Microsoft Azure SQL Data Warehouse.
Azure DW JDBC Password	Password to connect to the Microsoft Azure SQL Data Warehouse account.
Azure DW Schema Name	Name of the schema in Microsoft Azure SQL Data Warehouse.
Azure Storage Type	Type of Azure storage to stage the files. You can select any of the following storage type: - Azure Blob. Default. To use Microsoft Azure Blob Storage to stage the files. - ADLS Gen2. To use Microsoft Azure Data Lake Storage Gen2 as storage to stage the files.
Azure Blob Account Name	Name of the Microsoft Azure Storage account to stage the files.
Azure Blob Account Key	The key that authenticates the access to the Blob storage account.
ADLS Gen2 Storage Account Name	Name of the Microsoft Azure Data Lake Storage Gen2 account to stage the files.
ADLS Gen2 Account Key	Microsoft Azure Data Lake Storage Gen2 access key to stage the files.

Property	Description
Blob End-point	Type of Microsoft Azure endpoints. Select one of the following options: - `core.windows.net`: Default - `core.usgovcloudapi.net`: Select to access the US government Microsoft Azure Data Warehouse endpoints. - `core.chinacloudapi.cn`: Select to access a Microsoft Azure Data Warehouse endpoint in the China region. You can configure the US government Microsoft Azure end-points when a mapping runs in the native environment and on the Spark engine.
VNet Rule	Enable to connect to a Microsoft Azure SQL Data Warehouse endpoint residing in a virtual network (VNet).

Creating a Microsoft Azure SQL Data Warehouse Connection

Create a Microsoft Azure SQL Data Warehouse connection before you create a Microsoft Azure SQL Data Warehouse 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 **Database** > **Azure SQL Data Warehouse**, and click **Add**.
5. Enter a connection name and an optional description.
6. Select Azure SQL Data Warehouse as the connection type.
7. Click **Next**.
8. Configure the connection properties.
9. Click **Test Connection** to verify the connection to Microsoft Azure SQL Data Warehouse.
10. Click **Finish**.

CHAPTER 4

PowerExchange for Microsoft Azure SQL Data Warehouse Data Objects

This chapter includes the following topics:

Microsoft Azure SQL Data Warehouse Data Object Overview

A Microsoft Azure SQL Data Warehouse data object is a physical data object that represents Microsoft Azure SQL Data Warehouse table as a source or target. A Microsoft Azure SQL Data Warehouse data object is the representation of data that is based on a Microsoft Azure SQL Data Warehouse object.

You can configure the data object read and write operation properties that determine how data can be read from Microsoft Azure SQL Data Warehouse or loaded to Microsoft Azure SQL Data Warehouse.

To read data from the Microsoft Azure SQL Data Warehouse, create a data object read operation based on the Microsoft Azure SQL Data Warehouse data object. Configure the read operation properties to determine how the Data Integration Service must read data from the Microsoft Azure SQL Data Warehouse table. Add the read operation as a source in a mapping.

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

You can use A Microsoft Azure SQL Data Warehouse data object read operation to look up data in a Microsoft Azure SQL Data Warehouse table. You add the data object read operation to a mapping as a lookup transformation. You can look up data from a Microsoft Azure SQL Data Warehouse table in a mapping based on a lookup condition. You can configure an uncached lookup operation in the native environment.

Microsoft Azure SQL Data Warehouse Data Object Properties

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

General Properties

You can configure the following properties for a Microsoft Azure SQL Data Warehouse data object:

- Name. Name of the Microsoft Azure SQL Data Warehouse data object.
- Description. Description of the Microsoft Azure SQL Data Warehouse data object.
- Connection. Name of the Microsoft Azure SQL Data Warehouse connection.

IDENTITY Column

You can use the IDENTITY column in mappings. Consider the following instructions when you use the IDENTITY column:

- You can connect an IDENTITY column for source and lookup objects.
- You cannot connect the IDENTITY column for target objects.
- PowerExchange for Microsoft Azure SQL Data Warehousing does not highlight the IDENTITY column in the imported objects.

Microsoft Azure SQL Data Warehouse Data Object Read Operation

The Data Integration Service reads data from a Microsoft Azure SQL Data Warehouse table 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 SQL Data Warehouse table.

The Output transformation represents the data that the Data Integration Service passes into the mapping pipeline. Select the Output transformation to edit the ports, sources, query, run-time, and advanced properties.

You can specify the location of the staging directory, and securely write the results to Microsoft Azure SQL Data Warehouse.

Microsoft Azure SQL Data Warehouse Data Object Read Operation Properties

The Data Integration Service reads data from a Microsoft Azure SQL Data Warehouse object based on the data object read operation. The Developer tool displays the data object read operation properties of the Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse object. Select the source properties to view data, such as the name and description of the Microsoft Azure SQL Data Warehouse 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 set advanced properties, such as Azure Blob Container Name, Field Delimiter, and Number of Concurrent Connections to Blob Store.

Source Properties of the Data Object Read Operation

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

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

General Properties - Source

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

Property	Description
Name	Name of the Microsoft Azure SQL Data Warehouse source object.
Description	Description of the data object read operation.

Ports Properties - Source

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.
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.

Query Properties

Use the **Query** tab to specify a native expression filter condition. PowerExchange for Microsoft Azure SQL Data Warehouse does not support platform expression filter.

The following table describes the query properties that you can configure for Microsoft Azure SQL Data Warehouse sources:

Property	Description
Join	Not applicable
Filter	Filter value in a read operation. The filter specifies the WHERE clause of the SELECT statement. Use a filter to reduce the number of rows that the Data Integration Service reads from the Microsoft Azure SQL Data Warehouse source. When you enter a source filter, the Developer tool adds a WHERE clause to the default query. You can configure only native expression filters.

Run-time Properties

The **Run-time** tab displays the name of the connection that the Data Integration Service uses to read data from the Microsoft Azure SQL Data Warehouse. You can select a different connection or you can parameterize the connection. You can also configure key range partitioning. When you configure key range partitioning, the Data Integration Service distributes rows of data based on a port or set of ports that you define as the partition key.

Key Range Partitioning

When you configure the key range partitioning, the Data Integration Service distributes rows of source data based on the fields that you define as partition keys. The Data Integration Service compares the field value to the range values for each partition and sends rows to the appropriate partitions.

Use key range partitioning for columns that have an even distribution of data values. Otherwise, the partitions might have unequal size. For example, a column might have 10 rows between key values 1 and 1000 and the

column might have 999 rows between key values 1001 and 2000. If the mapping includes multiple sources, use the same number of key ranges for each source.

When you define key range partitioning for a column, the Data Integration Service reads the rows that are within the specified partition range.

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

- Bigint
- Bit
- Decimal
- Float
- Int
- Money
- Real
- Smallint
- Smallmoney
- Tinyint

After you configure key range partitioning, you must configure the maximum parallelism.

Configuring Maximum Parallelism for Microsoft Azure SQL Data Warehouse Mappings

You can configure maximum parallelism in the Administrator tool and in the Develop tool.

Mandatory. Ask the Informatica administrator to set the maximum parallelism for the Data Integration Service to a value equal to or greater than the number of partitions defined for the source object.

Maximum parallelism determines the maximum number of parallel threads that process a single pipeline stage. The Informatica Administrator can increase the **Maximum Parallelism** property value based on the number of CPUs available on the nodes where mappings run.

Set a maximum parallelism value for a mapping in the Developer tool.

By default, the **Maximum Parallelism** property for each mapping is set to **Auto**. Each mapping uses the maximum parallelism value defined for the Data Integration Service.

Set the maximum parallelism value in the mapping run-time properties to a value equal to or greater than the number of partitions defined for the source object. When you set the maximum parallelism to different integer values for the Data Integration Service and the mapping, the Data Integration Service uses the minimum value of the two.

Advanced Properties - Source

The advanced properties display the physical name of the Microsoft Azure SQL Data Warehouse object.

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

Property	Description
Azure Blob Container Name	Microsoft Azure Storage container name Required if you select Azure Blob storage in the connection properties. You must mention the Microsoft Azure Storage Container name to successfully run the mapping. When you mention the container name that does not exist, the data preview fails but the Data Integration Service creates a container with the specified name and successfully runs the mapping.
ADLS FileSystem Name	The name of the file system in Microsoft Azure Data Lake Storage Gen2. Required if you select ADLS Gen2 storage in the connection properties. You can also provide the path of the directory under given file system.
Staging File Format	Format of the staging files. You stage files in delimited or Parquet format.
Field Delimiter	Character used to separate fields in the file. Default is `0x1e`. The mapping fails when you parameterize a multiple delimiter. You can use single delimiter when you parameterize a delimiter. Ensure that the source or target data does not contain the delimiter character, new line character, or double quotes.
Number of Concurrent Connections to Blob Store	Number of concurrent connections to extract data from the Microsoft Azure Blob Storage. Default is 4.
Pre SQL*	Pre-SQL command that must be run before reading data from the source.
Post SQL*	Post-SQL command that must be run after writing data to the target.
On Pre-Post SQL Error	Required if the mapping uses pre- or post-session SQL. If you select Stop, the Data Integration Service stops the mapping errors executing pre-session or post-session SQL. If you select Continue, the Data Integration Service continues the mapping regardless of errors executing pre-session or post-session SQL. By default, the Data Integration Service stops the mapping upon pre- or post-session SQL error and marks the mapping failed.
SQL Override*	When you read data from a Microsoft Azure SQL Data Warehouse object, you can configure SQL overrides and define constraints.
Quote Character*	The Data Integration Service skips the specified character when you read data from Microsoft Azure SQL Data Warehouse.
Schema Name	Overrides the schema specified in the connection.
Table Name	Overrides the table name of the imported Microsoft Azure SQL Data Warehouse source table.
**These properties are not applicable to an uncached lookup transformation.*	

Microsoft Azure SQL Data Warehouse Data Object Write Operation Properties

The Data Integration Service writes data to a Microsoft Azure SQL Data Warehouse object based on the data object write operation. The Developer tool displays the data object write operation properties for the Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse 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 SQL Data Warehouse. Select the target properties to view data, such as the name and description of the Microsoft Azure SQL Data Warehouse object.

Input Properties of the Data Object Write Operation

Input properties represent data that the Data Integration Service writes to a Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse objects.

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

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

Ports Properties - Input Write

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

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

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

Run-time Properties

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

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

Property	Description
Connection	Name of the Microsoft Azure SQL Data Warehouse connection.
Partition Type	Not applicable. By default, the pass-through partition is applied to the Microsoft Azure SQL Data Warehouse target object.

Advanced Properties - Target

The advanced properties allow you to specify data object write operation properties to write data to a Microsoft Azure SQL Data Warehouse server.

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

Property	Description
Azure Blob Container Name	Microsoft Azure Storage container name. Required if you select Azure Blob storage in the connection properties. You must mention the Microsoft Azure Storage Container name to successfully run the mapping. When you mention a container name that does not exist, Data Integration Service creates a container with the specified name and successfully runs the mapping.
ADLS FileSystem Name	The name of the file system in Microsoft Azure Data Lake Storage Gen2. Required if you select ADLS Gen2 storage in the connection properties. You can also provide the path of the directory under given file system.
Field Delimiter	Character used to separate fields in the file. Default is `0x1e`. The mapping fails when you parameterize a multiple delimiter. You can use single delimiter when you parameterize a delimiter. Ensure that the source or target data does not contain the delimiter character, new line character, or double quotes.
Number of Concurrent Connections to Blob Storage	Number of threads to use to move data to the staging area in Microsoft Azure Blob Storage. Default is 4.
Truncate Table	Specify the truncate table property when you use insert operations. Truncates the target before inserting data to the target.
Pre SQL	Pre-SQL command that must be run before reading data from the source.
Post SQL	Post-SQL command that must be run after writing data to the target.

Property	Description
On Pre-Post SQL Error	Required if the mapping uses pre- or post-session SQL. If you select Stop, the Data Integration Service stops the mapping errors executing pre-session or post-session SQL. If you select Continue, the Data Integration Service continues the mapping regardless of errors executing pre-session or post-session SQL. By default, the Data Integration Service stops the mapping upon pre- or post-session SQL error and marks the mapping failed.
Quote Character	Specifies the quote character to skip when you write data to Microsoft Azure SQL Data Warehouse. When you write data to Microsoft Azure SQL Data Warehouse and the source table contains the specified quote character, the task fails. Change the quote character value to a value that does not exist in the source table.
Treat Row Source As	Select one of the following options: - INSERT. Inserts rows. By default, the insert operation is enabled. - DELETE. Deletes rows based on the primary key defined. - UPDATE. Updates rows based on the primary key defined. - UPSERT: Updates the existing rows and inserts the remaining rows. **Note:** To perform a delete, update, or upsert operation, you must map a key column in the **Keys** view of the data object.
Compression Format	Compresses the staging files in the `.Gzip` format in the native environment. Default is None.
Reject Threshold	Number of errors within a batch that causes a batch to fail. Enter a positive integer. If the number of errors is equal to or greater than the property value, the Integration Service rejects the entire batch to the error file and marks the session failed. Default is 1.
Schema Name	Overrides the schema specified in the connection for an existing target.
Table Name	Overrides the table name of the existing Microsoft Azure SQL Data Warehouse target table.
Batch Size	Minimum number of rows in a batch. Enter a number greater than 0. Applicable to the native environment. Default is 2000000.
Target Schema Strategy	Target schema strategy for the Azure SQL Data Warehouse target table. You can select one of the following target schema strategies: - RETAIN - Retain existing target schema - CREATE - Create or replace table at run time - Assign Parameter **Note:** Applicable when you run a mapping in the native environment, on the Spark engine, or on the Databricks Spark engine. For more information about Target Schema Strategy, see "Target Schema Strategy" on page 33.

Rejected Rows

The Integration Service generates two error files in the Microsoft Azure Blob container specified in the target properties.

One error file contains an entry for each rejected row and the other error file lists the cause for the rejected rows. To generate the error files, specify **Reject Threshold** in target properties.

Importing a Microsoft Azure SQL Data Warehouse Data Object

Import a Microsoft Azure SQL Data Warehouse 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 **Azure SQL Data Warehouse Data Object** and click **Next**.

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

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

Creating Keys in a Microsoft Azure SQL Data Warehouse Data Object

Create key columns to identify each row in a data object. The key column is used to perform delete, update, and upsert operations.

1. Open a Microsoft Azure SQL Data Warehouse data object.
2. Select the Keys view.
3. Click **Add**.
4. Select a key column from the available columns list.
5. Click **OK**.
6. Save the data object.

Creating a Microsoft Azure SQL Data Warehouse Data Object Read or Write Operation

You can add a Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse data objects.

Before you create a Microsoft Azure SQL Data Warehouse data object read or write operation, you must create at least one Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse 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 SQL Data Warehouse Target

You can create a new Microsoft Azure SQL Data Warehouse target based on the Source transformation or any transformation.

1. Select a project or folder in the **Object Explorer** view.
2. Select a source or a transformation in the mapping.
3. Right-click and select **Create Target**.
   The **Create Target** dialog box appears.
4. Select **Others** and then select **AzureDW Data Object** from the list in the **Data Object Type** section.
5. Click **OK**.
   The **New AzureDW Data Object** dialog box appears.
6. Enter a name for the data object.
7. Click **Finish**.

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

Note: Create Target is not applicable when you use an ODBC connection to connect to Microsoft Azure SQL Data Warehouse.

Rules and Guidelines for Creating a Microsoft Azure SQL Data Warehouse Target

Use the following rules and guidelines when you create a new Microsoft Azure SQL Data Warehouse target:

- You must select the connection for the new target data object in the **Connection** property after you create a new target. You must also provide the blob container name in the write operation data object properties.
- The Data Integration Service creates the new target table with the name specified in the **Native Name** property in the **Data Object Details**.
- When you create a new target, the Data Integration Service does not create any constraints or keys defined in the source table in the target table.
- When you create a new target, the Data Integration Service creates all the columns of the target table with NULL constraint.
- When you create a new target, the Data Integration Service creates columns with the following data types in the target table:

Source Table Data Type	Target Table Data Type
binary, varbinary	varbinary
bit, int, smallint, tinyint	int
char, datetime2, datetimeoffset, nchar, nvarchar, time, varchar	nvarchar
date, datetime, smalldatetime	datetime
decimal, money, smallmoney	decimal
float, real	float
bigint	bigint

- When a Microsoft Azure SQL Data Warehouse table contains a binary field and you select **Create Target** to create a complex file, the mapping fails on the Spark and Databricks Spark engines. To run the mapping successfully, ignore the binary field and include rest of the fields as required when you select **Create Target**.
- You cannot use the table name override property to create a new table.

CHAPTER 5

Microsoft Azure SQL Data Warehouse Mappings

This chapter includes the following topics:

Microsoft Azure SQL Data Warehouse Mapping Overview

After you create a Microsoft Azure SQL Data Warehouse data object operation, you can develop a mapping.

You can define the following types of objects in the mapping:

- A Read transformation of the Microsoft Azure SQL Data Warehouse data object to read data from Microsoft Azure SQL Data Warehouse.
- A Write transformation of the Microsoft Azure SQL Data Warehouse data object to write data to Microsoft Azure SQL Data Warehouse.

Validate and run the mapping. You can deploy the mapping and run it or add the mapping to a Mapping task in a workflow.

When an Azure table contains a bad record, the Data Integration Service fails the mapping instead of rejecting the bad record.

When you run a mapping on the Azure HDInsight cluster and use Microsoft Azure Data Lake Storage Gen2 to stage the files, you cannot use a private endpoint to connect to Microsoft Azure Data Lake Storage Gen2.

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.

In an existing mapping, the real and float data types fields are imported as the decimal data type. After you upgrade, for the real and float data types, change the decimal data type to double only if you want to run the mapping on the Spark engine. Change the data type to double for all operations used in the mapping.

Audit

To validate the consistency and accuracy of data processed in a mapping for a read operation, you can create an audit for the mapping.

An audit is composed of rules and conditions. Use a rule to compute an aggregated value for a single column of data. Use a condition to make comparisons between multiple rules or between a rule and constant values.

You can run audits with mappings that run on the Data Integration Service or the Spark engine.

When you use SQL Override in the source on the Data Integration Service, ensure that any one of the following conditions is true:

- All the fields in the source and target are connected in the same order.
- The audit rule is added for all the columns specified in the SQL override query.

For more information, see the *Data Engineering Integration 10.5 User Guide*.

Microsoft Azure SQL Data Warehouse Mapping Example

Sales department of your organization has a large amount of customer details for all regions in flat files. The sales analyst needs to analyze the customer data in a given region in a short span of time. Create a mapping that reads all the customer records and write the records to Microsoft Azure SQL Data Warehouse table.

You can use the following objects in a Microsoft Azure SQL Data Warehouse 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 transformation.

Transformations

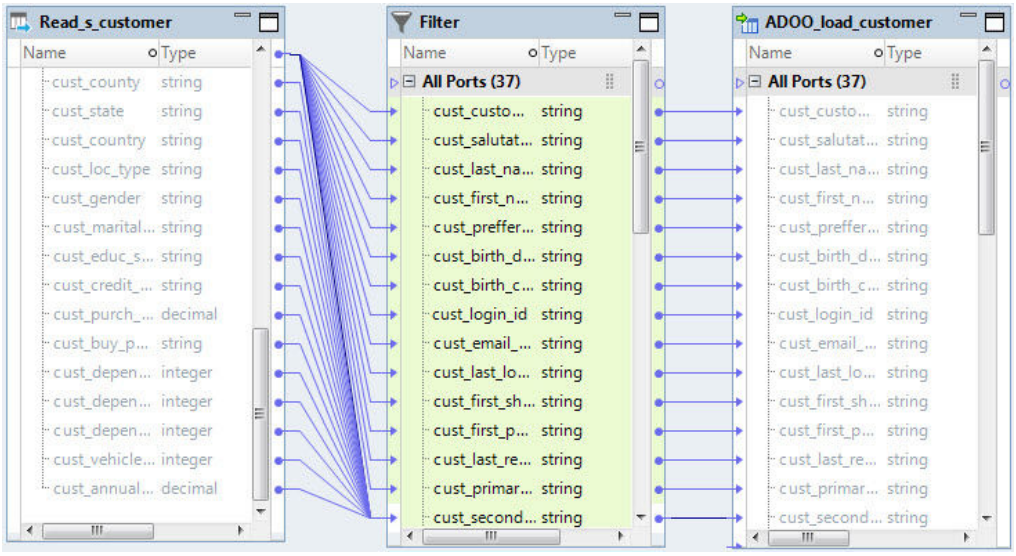
Add Filter transformation to get customer data in a particular region.

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 SQL Data Warehouse output

Create a Microsoft Azure SQL Data Warehouse data object write operation. Configure the Microsoft Azure SQL Data Warehouse connection and specify the Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse mapping example:



When you run the mapping, the customer records are read from the flat file and written to the Microsoft Azure SQL Data Warehouse table.

Microsoft Azure SQL Data Warehouse Dynamic Mapping Overview

You can use Microsoft Azure SQL Data Warehouse data objects as dynamic sources and targets in a mapping.

Use the Microsoft Azure SQL Data Warehouse dynamic mapping to accommodate changes to source, target, and transformation logics at run time. You can use a Microsoft Azure SQL Data Warehouse 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.

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.

Note: If refresh schema is enabled for a target transformation, you cannot perform upsert, update, and delete operations.

Even though the original order of the source or target ports in the table changes, the Data Integration Service displays the original order of the ports in the table 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*.

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.

Target Schema Strategy

You can choose to retain an existing target table or create a new target table in the target when you run a dynamic mapping.

You can select one of the following options in the **Target Schema Strategy** advanced properties for the data object write operation:

RETAIN - Retain existing target schema

The Data Integration Service retains the existing target schema.

Note: When you select **RETAIN** option and if the target table does not exist or the metadata of the source and target tables do not match, the mapping fails.

CREATE - Create table at run time

The Data Integration Service creates a new table based on the data object or the mapping flow if the table does not exist in the target.

When the Data Integration Service creates a table based on the data object, the table contains columns that match the ports in the data object. When the Data Integration Service creates a table based on the mapping flow, the table contains columns that match generated ports in the Write transformation.

Note: When you select the **CREATE** option and if the table already exists in the target location, the data is appended to the existing target table. To replace the existing table with the new table, create a dummy mapping and specify the DROP TABLE command in Pre SQL to delete the existing table. Before running the actual mapping to create a target at runtime, run the dummy mapping to delete the existing table. You cannot specify the DROP TABLE command in Pre SQL of the same mapping where you enable Create Target. Use the following syntax for the DROP TABLE command:

```
DROP TABLE <tableSchemaName>.<tableName>
```

Assign Parameter

You can assign a parameter to represent the value for the target schema strategy and then change the parameter at run time.

Rules and Guidelines for Microsoft Azure SQL Data Warehouse Dynamic Mappings

Consider the following rules and guidelines for Microsoft Azure SQL Data Warehouse dynamic mappings:

- You cannot override table name in a dynamic mapping.

- When you stage files in parquet format on the Spark engine or Databricks Spark engine, ensure that the source table does not have a column named FileName. The name is case insensitive.
- When you stage files in parquet format on the Spark engine or Databricks Spark engine, ensure that the source table does not have `datetime2` data.

Microsoft Azure SQL Data Warehouse 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 SQL Data Warehouse 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 **AzureDW 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 **AzureDW 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.

CHAPTER 6

Pushdown Optimization

This chapter includes the following topics:

Pushdown Optimization Overview

You can use pushdown optimization to push transformation logic to source databases or target databases. Use pushdown optimization to improve task performance by using the database resources. When you run a task configured for pushdown optimization, the task converts the transformation logic to an SQL query. The task sends the query to the database, and the database executes the query.

Example: You work for a rapidly growing data science organization. Your organization develops software products to analyze financials, building financial graphs connecting people profiles, companies, jobs, advertisers, and publishers. The organization uses infrastructure based on Microsoft Azure Services and stores its data in Microsoft Azure SQL Data Warehouse, a petabyte-scale data warehouse. The organization plans to implement a business intelligence service to build visualization and perform real-time analysis. Therefore, you need to port the vast amount of data stored in one database of Microsoft Azure SQL Data Warehouse to another Microsoft Azure SQL Data Warehouse database. And, then use MPP to run high-performance analytics.

You can use an ODBC connection to read this large amount of data from and write data to Microsoft Azure SQL Data Warehouse. Use full pushdown for the ODBC connection type to enhance the performance.

To read data from and write data to a Microsoft Azure SQL Data Warehouse object using the OBDC connection, perform the following steps:

1. Download and install the Microsoft ODBC driver.
2. Create a system DSN for Microsoft Azure SQL Data Warehouse.
3. Import the Microsoft Azure SQL Data Warehouse source and target objects.
4. Create an ODBC connection to access the Microsoft Azure SQL Data Warehouse source and target objects.

5. Create and run a mapping.

Install the ODBC Driver

Before you establish an ODBC connection to connect to Microsoft Azure SQL Data Warehouse, you must install the Microsoft ODBC Driver 13 for SQL Server for Windows. Install the *Microsoft ODBC Driver 13 for SQL Server 32-bit* on the Developer Client machine and the *Microsoft ODBC Driver 13 for SQL Server 64-bit* on the Data Integration Service host machine.

Download the Microsoft ODBC driver from the following link:
https://www.microsoft.com/en-us/download/details.aspx?id=53339

Create a System DSN

Before creating an ODBC connection, create a system DSN on the Developer Client machine and the Data Integration Service host machine.

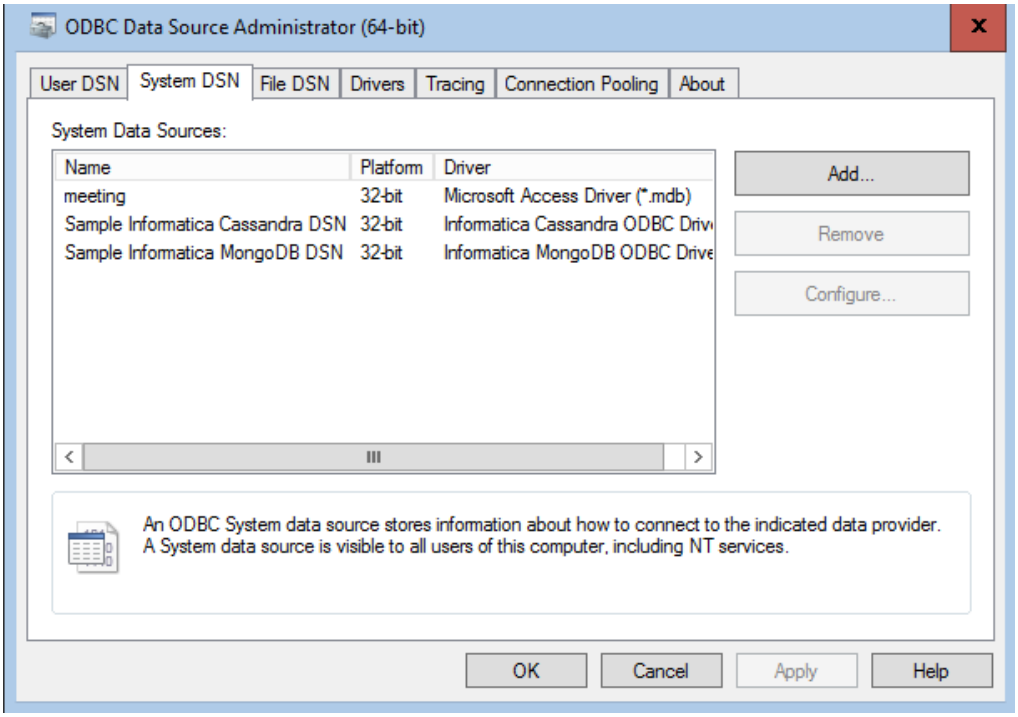
Perform the following steps to create a system DSN on the Developer Client:

1. Double-click the `odbcad32.exe` file under `C:\WINDOWS\SysWOW64`.

   The **ODBC Data Sources Administrator** box appears.

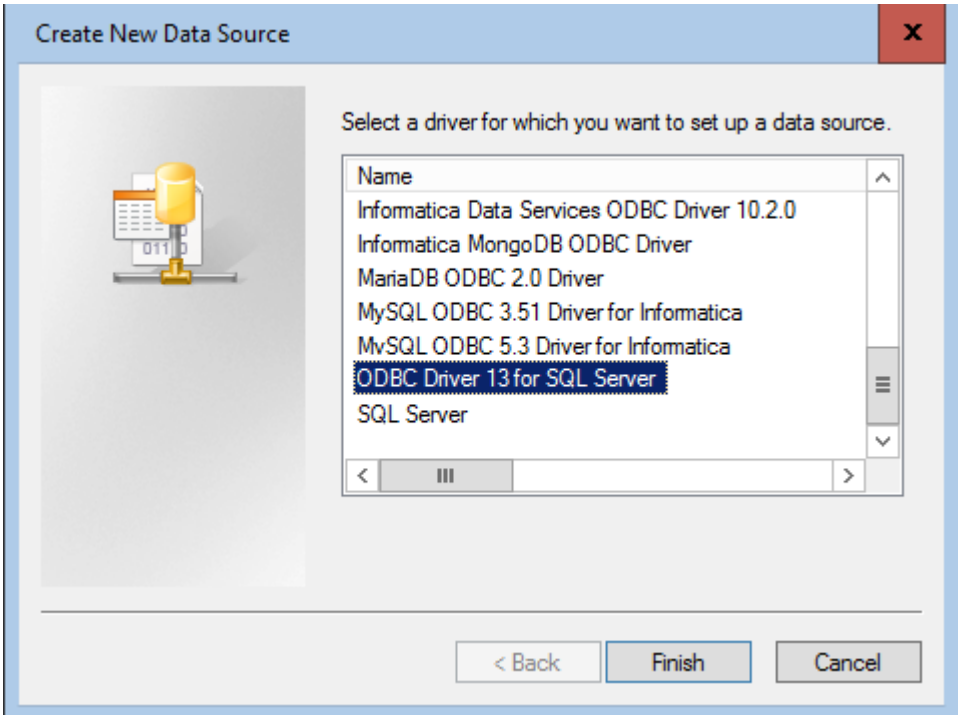
2. Click **System DSN**.

   The **System DSN** tab appears. The following image shows the **System DSN** tab on the **ODBC Data Sources Administrator** box:



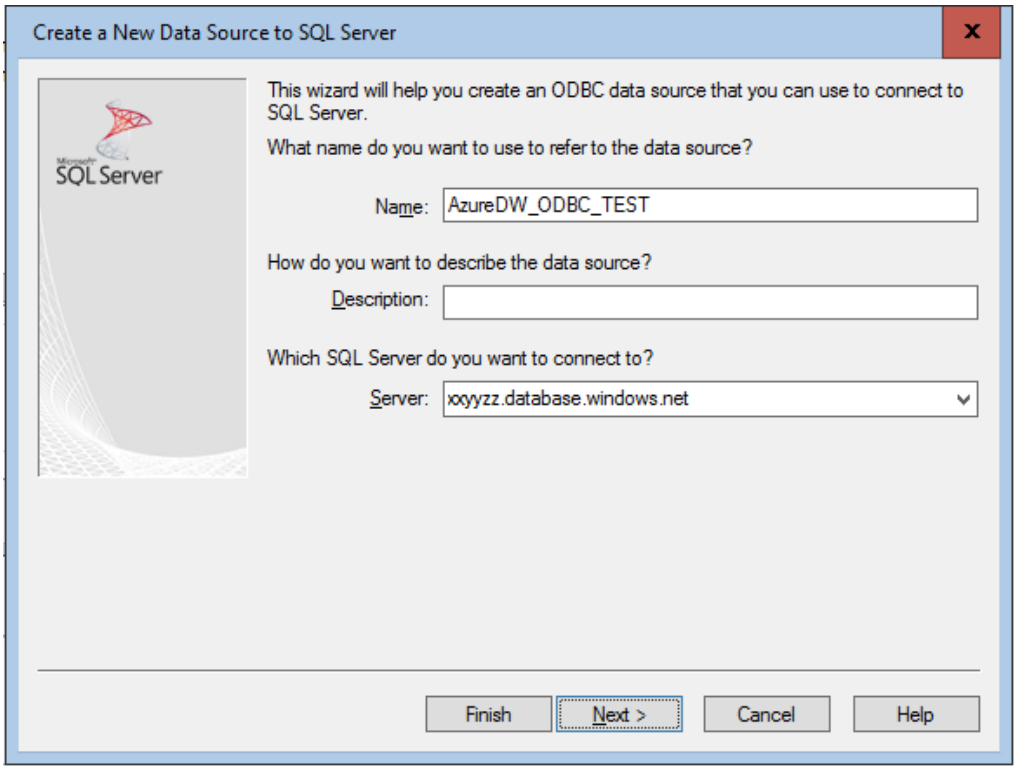
3. Click **Add**.

   The **Create New Data Source** box appears.



4. Select **ODBC Driver 13 for SQL Server** and click **Finish**.

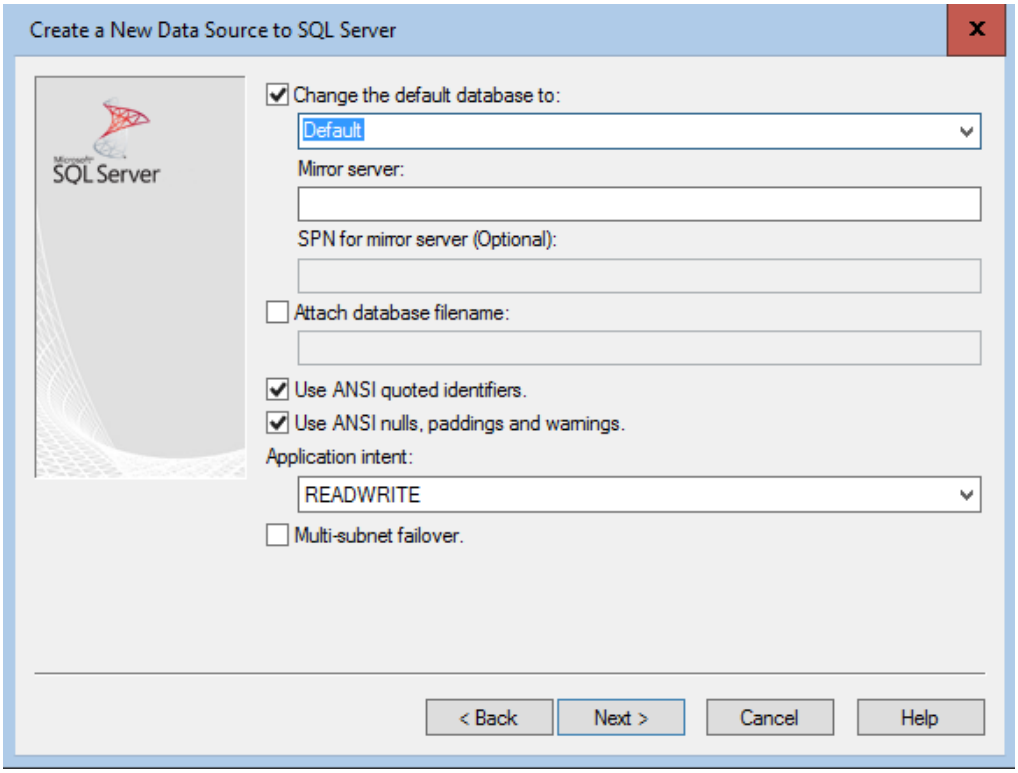
   The Create a New Data Source to SQL Server box appears.



5. Specify the name, description, and the server you want to connect to. The following table lists the sample DSN names on the Developer Client machine and the Data Integration Service host machine:

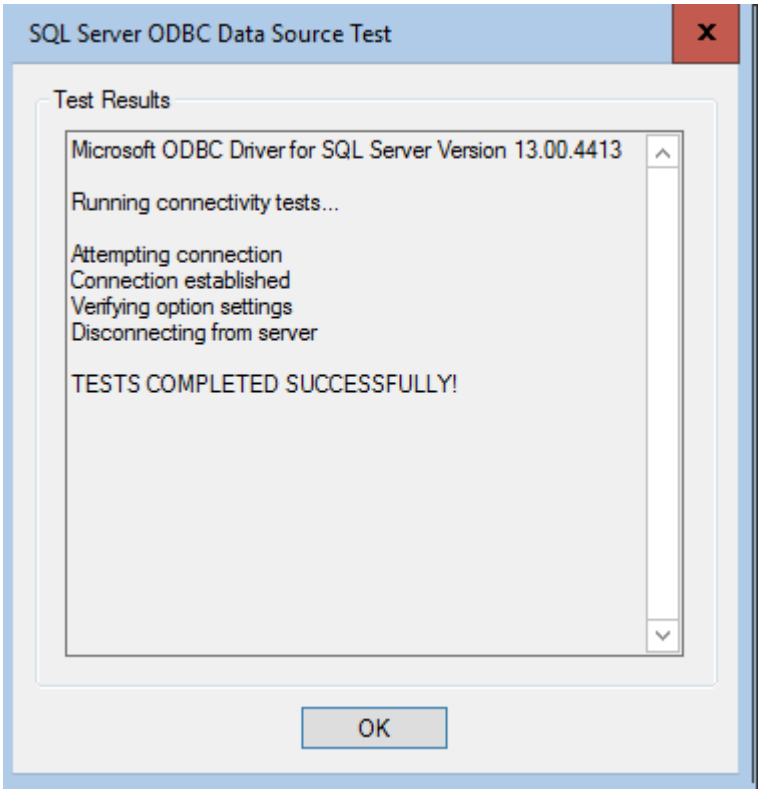
Machine	DSN
Developer Client machine	AzureDW_ODBC_TEST
Data Integration Service host machine	AzureDW_ODBC_SERVER

6. Click **Next** and provide the user name and password for SQL Server Authentication.
7. Click **Next** and change the default database setting to the database you want to connect to.



8. Click **Next**.

9. Click **Finish** and then, click **Test Data Source**. Ensure that the test connection completes successfully on the Developer Client machine and the Data Integration Service host machine.



10. Click **OK**.

Similarly, you can create a system DSN on the Data Integration Service host machine. Double-click the `odbcad32.exe` file under `C:\WINDOWS\system32` on the Data Integration Service host machine.

Import the Microsoft Azure SQL Data Warehouse Source and Target Objects

Import the Microsoft Azure SQL Data Warehouse source object from which you want to read data and the target object to write data.

1. Select a project or folder in the Object Explorer view.
2. Click **File** > **New** > **Data Object**.
3. Select **Relational Data Object** and click **Next**.
   The **New Relational Data Object** dialog box appears.
4. Click **Browse** next to the Connection option and select the ODBC connection from which you want to import the Microsoft Azure SQL Data Warehouse resources.
5. Click **Create data from existing resource**.
6. To add a resource to the Relational Data Object, click **Browse** next to the Resource option.
7. Navigate to the resource to add it to the data object and click **Ok**.

8. Click **Browse** next to the Location option and select the target project or folder.
9. Click **Finish**.

The data object appears under Data Object in the project or folder in the **Object Explorer** view. You can also add resources to a relational data object after you create it.

Create an ODBC Connection

Create ODBC connections to access Microsoft Azure SQL Data Warehouse source and target objects.

1. In the Developer Client, click **New Connections**.

   The **Select a Connection Category** box appears.
2. Select **Connections**.
3. Specify the name, description, and location for the connection.
4. Select **Type** as **ODBC** and click **Next**.
5. Configure the following connection properties:

Relational Connection Property	Description
User Name	Enter the user name to connect to the Microsoft Azure SQL Data Warehouse database.
Password	Enter the password to connect to the Microsoft Azure SQL Data Warehouse database.
Connection String	Enter the name of the ODBC data source that you created for Microsoft Azure SQL Data Warehouse database on the Data Integration Service host machine. For example: AzureDW_ODBC_SERVER
Code Page	Select the code page that the Data Integration Service uses to read or write data.
ODBC Provider	Select **AzureDW**.

6. Test the connection and click **OK**.

   The Microsoft Azure SQL Data Warehouse ODBC connection is created successfully.

Create a Mapping

After you import the source and target objects, create a mapping to read data from and write data to Microsoft Azure SQL Data Warehouse.

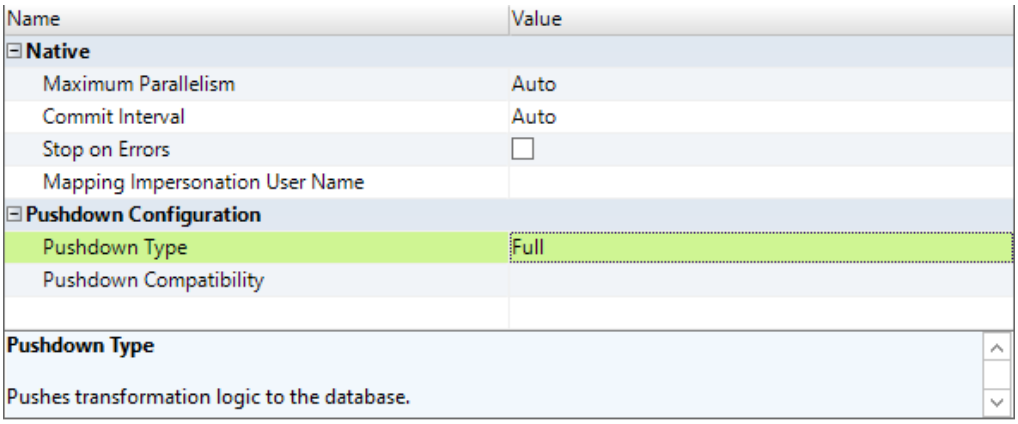
In a mapping, you define properties that determine how the Data Integration Service extracts data from or loads data to a data source. Configure pushdown optimization for the source and target objects.

1. Select a project or folder in the Object Explorer view.
2. Click **Tasks** > **Create**.
3. Click **File** > **New** > **Mapping**.

4. Enter a mapping name and click **Finish**.

   An empty mapping appears in the editor.
5. Drag a data object to the editor and select Read to add the data object as a source.
6. Drag a data object to the editor and select Write to add the data object as a target.
7. On the **Properties** tab, select **Run-time**.
8. Select `Full` as pushdown type.

   The following image shows the pushdown optimization configuration:



9. Save and run the mapping.

Supported Pushdown Optimization Functions

PowerExchange for Microsoft Azure SQL Data Warehouse supports full pushdown optimization.

The following table summarizes the availability of pushdown functions in a Microsoft Azure SQL Data Warehouse. Columns marked with an X indicate that the function can be pushed to the Microsoft Azure SQL Data Warehouse by using full pushdown optimization. Columns marked with a dash (-) symbol indicate that the function cannot be pushed to the database.

Function	Pushdown	Function	Pushdown	Function	Pushdown
ABORT()	-	INITCAP()	-	REG_MATCH()	-
ABS()	X	INSTR()	X	REG_REPLACE	-
ADD_TO_DATE()	-	IS_DATE()	-	REPLACECHR()	-
AES_DECRYPT()	-	IS_NUMBER()	-	REPLACESTR()	-
AES_ENCRYPT()	-	IS_SPACES()	-	REVERSE()	-
ASCII()	X	ISNULL()	X	ROUND(DATE)	-
AVG()	X	LAST()	-	ROUND(NUMBER)	X
CEIL()	X	LAST_DAY()	-	RPAD()	-

Function	Pushdown	Function	Pushdown	Function	Pushdown
CHOOSE()	-	LEAST()	-	RTRIM()	X
CHR()	X	LENGTH()	X	SET_DATE_PART()	-
CHRCODE()	-	LN()	-	SIGN()	X
COMPRESS()	-	LOG()	X	SIN()	X
CONCAT()	X	LOOKUP	-	SINH()	X
COS()	X	LOWER()	X	SOUNDEX()	X
COSH()	-	LPAD()	-	SQRT()	X
COUNT()	X	LTRIM()	X	STDDEV()	X
CRC32()	-	MAKE_DATE_TIME()	-	SUBSTR()	X
CUME()	-	MAX()	X	SUM()	X
DATE_COMPARE()	X	MD5()	-	SYSDATE()	X
DATE_DIFF()	-	MEDIAN()	-	SYSTIMESTAMP()	X
DECODE()	X	METAPHONE()	-	TAN()	X
DECODE_BASE64()	-	MIN()	X	TANH()	X
DECOMPRESS()	-	MOD()	X	TO_BIGINT	X
ENCODE_BASE64()	-	MOVINGAVG()	-	TO_CHAR(DATE)	X
EXP()	X	MOVINGSUM()	-	TO_CHAR(NUMBER)	X
FIRST()	-	NPER()	-	TO_DATE()	X
FLOOR()	X	PERCENTILE()	-	TO_DECIMAL()	X
FV()	-	PMT()	-	TO_FLOAT()	X
GET_DATE_PART()	X	POWER()	X	TO_INTEGER()	X
GREATEST()	-	PV()	-	TRUNC(DATE)	-
IIF()	X	RAND()	-	TRUNC(NUMBER)	X
IN()	-	RATE()	-	UPPER()	X
INDEXOF()	-	REG_EXTRACT()	-	VARIANCE()	X

The following table lists the pushdown operators that can be used in a Microsoft Azure SQL Data Warehouse. Columns marked with an X indicate that the operator can be pushed to the Microsoft Azure SQL Data

Warehouse by using full pushdown optimization. Columns marked with a dash (-) symbol indicate that the operator cannot be pushed to the database.

Operator	Pushdown
+ - * /	X
%	X
||	-
= > < >= <= <>	X
!=	X
^=	X
NOT AND OR	X

The following table summarizes the availability of variables in Microsoft Azure SQL Data Warehouse. Columns marked with an X indicate that the variable can be pushed to the Microsoft Azure SQL Data Warehouse by using full pushdown optimization. Columns marked with a dash (-) symbol indicate that the variable cannot be pushed to the Microsoft Azure SQL Data Warehouse.

Variable	Pushdown
SESSSTARTTIME	X
SYSDATE	X
WORKFLOWSTARTTIME	-

PowerExchange for Microsoft Azure SQL Data Warehouse supports the following transformations for full pushdown:

- Aggregator transformation
- Expression transformation
- Filter transformation
- Joiner transformation
- Lookup transformation
- Router transformation
- Union transformation

Rules and Guidelines for Functions in Pushdown Optimization

Use the following rules and guidelines when pushing functions to a Microsoft Azure SQL Data Warehouse database:

- Upsert operations for pushdown optimization are not applicable when you use an ODBC connection.

- The Microsoft Azure SQL Data Warehouse aggregate functions accept only one argument, which is a field set for the aggregate function. The agent ignores any filter condition defined in the argument. In addition, ensure that all fields mapped to the target are listed in the `GROUP BY` clause.
- You cannot push the `TO_BIGINT()` or `TO_INTEGER()` function with more than one argument to the Microsoft Azure SQL Data Warehouse database.
- To push `INSTR()` to Microsoft Azure SQL Data Warehouse database, you must only define string, search_value, and start arguments. Microsoft Azure SQL Data Warehouse does not support occurrence and comparison_type arguments.
- When you push the `TO_CHAR(NUMBER)` function to the Microsoft Azure SQL Data Warehouse database, the number rounds off to a precision of 6.
- When you push the `SYSTIMESTAMP()` and `SYSDATE()` functions to the Microsoft Azure SQL Data Warehouse database, do not specify any format. The Microsoft Azure SQL Data Warehouse database returns the complete time stamp. `SYSDATE` works without brackets () only, if used it shows as invalid expression.
- To push datetime formats, change the default date format to `MM/DD/YYYY` in **Preferences** > **Run Configurations** > **Mapping** > **Advanced** > **Default date time format** in the Developer Client.
- To push the `TO_CHAR()` function to the Microsoft Azure SQL Data Warehouse database, you must define the date and format arguments or else the default date format given in **Preferences** is used.
- To push `TO_CHAR(DATE)` and `TO_DATE()` functions to the Microsoft Azure SQL Data Warehouse database, use the following formats for date data types:

  - `MON DD YYYY HH12:MI AM`
  - `MM/DD/YY, DD/MM/YY, YY/MM/DD`
  - `MM/DD/YYYY, DD/MM/YYYY, YYYY/MM/DD`
  - `YY.MM.DD, DD.MM.YY`
  - `YYYY.MM.DD, DD.MM.YYYY`
  - `DD-MM-YY, MM-DD-YY`
  - `DD-MM-YYYY, MM-DD-YYYY`
  - `DD MON YY, MON DD, YY`
  - `DD MON YYYY, MON DD, YYYY`
  - `HH:MI:SS`
  - `YYMMDD`
  - `YYYYMMDD`
  - `DD MON YYYY HH24:MI:SS:MS`
  - `HH24:MI:SS:MS`
  - `YYYY-MM-DD HH24:MI:SS`
  - `YYYY-MM-DD HH24:MI:SS.MS`
  - `YYYY-MM-DD (works only with TO_CHAR(DATE))`
  - `MM/DD/YY HH12:MI:SS AM (works only with TO_CHAR(DATE))`
- To push `GET_DATE_PART()` and `DATE_COMPARE()` functions to the Microsoft Azure SQL Data Warehouse database, use the following formats for date data types:

  - `D`
  - `DDD`
  - `HH24`

- MI
- MM
- MS
- SS
- YYYY

APPENDIX A

Microsoft Azure SQL Data Warehouse Datatype Reference

This appendix includes the following topics:

Datatype Reference Overview

Informatica Developer uses the following data types in Microsoft Azure SQL Data Warehouse mappings:

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

Microsoft Azure SQL Data Warehouse and Transformation Datatypes

The following table lists the Microsoft Azure SQL Data Warehouse data types that the Data Integration Service supports and the corresponding transformation data types:

Microsoft Azure SQL Data Warehouse Native Data Type	Transformation Data Type	Range
bigint	bigint	-9,223,372,036,854,770,000 to 9,223,372,036,854,770,000
binary	binary	1 to 8000 bytes
bit	integer	0,1,NULL
char	string	1 to 8000 characters
date	date/time	January 1, 1753 00:00:00 to 12/31/9999
datetime	date/time	January 1, 1753 00:00:00 to December 31, 9999 23:59:59.997
datetime2	string	1 to 8000 characters When you stage files in parquet format, the precision for datetime2 is 6.
datetimeoffset	string	
decimal	decimal	Precision 28, scale 0 to 5 **Note:** Data corruption is possible when the table has a defined decimal column with precision more than 28, but the table contains data less than 28.
float	double	Precision 7, scale 0 to 7
int	integer	-2,147,483,648 to 2,147,483,647
money	decimal	-922,337,203,685,477.0000 to 922,337,203,685,477.0000
nchar	string	1 to 4000 characters
nvarchar	string	1 to 4,000 characters
real	double	Precision 7, scale 0 to 7
smalldatetime	date/time	1/1/1900 0:00 to 6/6/2079 23:59
smallint	integer	-32,768 to 32,767
smallmoney	decimal	-214,748.3648 to 214,748.3647

Microsoft Azure SQL Data Warehouse Native Data Type	Transformation Data Type	Range
time	string	00:00:00.0000000 to 23:59:59.9999999
tinyint	integer	0 to 255 When you stage files in parquet format, the range is 0 to 127.
varbinary	binary	1 to 8000 bytes
varchar	string	1 to 8000 characters

Rules and Guidelines for Data Types

Consider the following rules and guidelines for data types:

- The `datetimeoffset` data type is supported only for passthrough mappings that read data from and write data to Microsoft Azure SQL Data Warehouse objects.
- Binary, Date, Datetime, Datetime2, Smalldatetime, and Varbinary data types are not supported in a lookup condition.
- Binary, Datetime, and Varbinary data types are not supported on the Databricks Spark engine.
- The Spark engine cannot process dates to the nanosecond. It can return a precision for date/time data up to the microsecond.
- After you upgrade from a version prior to 10.2.2 to 10.5.0, the existing mappings that contain the following data types fail on the Spark engine at run time:

  - Binary
  - Varbinary
  - Datetime2
  - Datetimeoffset

  To run the existing mappings successfully, you must map these data types to the string data type or re-import the object.

  - Binary -> String (n)
  - Varbinary -> String (n)
  - Datetime2 -> String (27)
  - Datetimeoffset -> String (34)

INDEX

T