



Best Practices Guide

Qlik Data Integration[®] and Databricks[®]

Prasad Kona | Databricks

Clever Anjos | Qlik

Tim Garrod | Qlik

Table of Contents

- Introduction.....2**
- Qlik Data Integration Solutions2**
 - Qlik Replicate.....2
 - Qlik Compose3
- Databricks Lakehouse Platform4**
- Qlik Data Integration Solutions for Databricks4**
 - 1. Before August / 2022 version Qlik Replicate Databricks Lakehouse (Delta) Endpoint4
 - 2. Qlik Replicate + Qlik Compose for non-Delta endpoints5
 - 3. Databricks Lakehouse (Delta) Connector (Available with Qlik Replicate and Qlik Compose version August 2022 or after)7
- Recommendations for Optimizing Qlik Replicate CDC with Databricks Lakehouse (Delta) connectors.8**
 - 1. Databricks runtime version supporting Photon8
 - 2. Select “Memory optimized – Delta cache accelerated”9
 - 3. Configure Auto-Optimize options10
 - 4. Optimize tables on a regular basis10
 - 5. Table selections in a task10
 - 6. Auto scaling.....11
 - 7. File size configuration11
 - 8. Batch Tuning settings12
 - 9. Changing from I/U/D to D/I (upsert mode in Qlik Replicate)14
 - 10. Partition large tables14
- Recommendations for Optimizing Qlik Replicate and Compose Model 15**
 - 11. Enable Speed Partitions15
 - 12. Partition Timeframe.....16
 - 13. Replicate file delivery.....17
 - 14. Compose processing interval17
 - 15. Latency impact.....17
 - 16. SEQUENCE as Target storage format18
- Appendix I - Creating Qlik Replicate Connections 19**
 - 1. Obtaining Databricks credentials and connection details19
 - 2. Databricks on AWS20
 - 3. Databricks on Google Cloud22
 - 4. Microsoft Azure Databricks.....23
 - 5. Microsoft Azure Databricks Delta25
 - 6. Databricks Lakehouse (Delta).....27

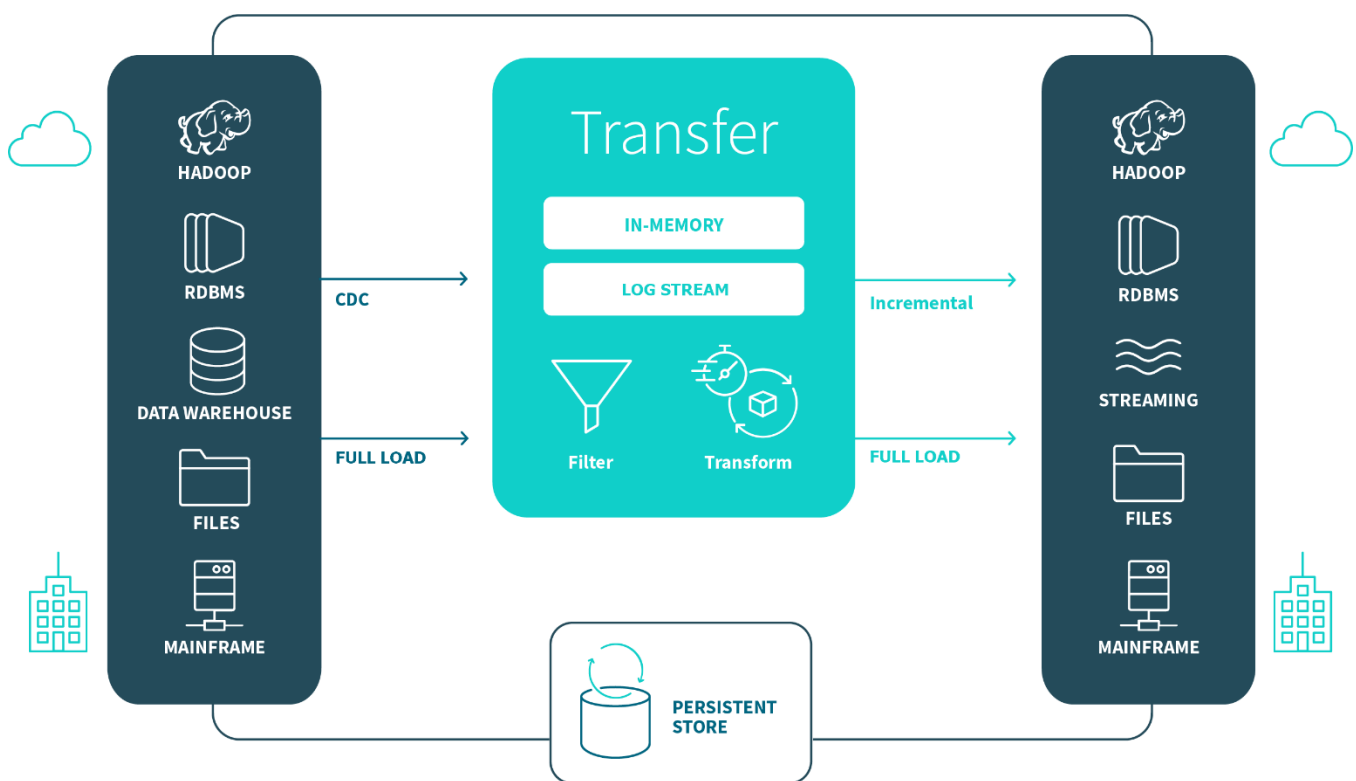
Introduction

Qlik® has a powerful real-time data pipeline automation solution for Databricks Lakehouse Platform customers. Qlik Data Integration solutions accelerate machine learning (ML), artificial intelligence (AI), and DataOps initiatives with Change Data Capture (CDC) technology that ensures continuous data streams from multiple data sources to the Databricks Lakehouse Platform.

Qlik Data Integration Solutions

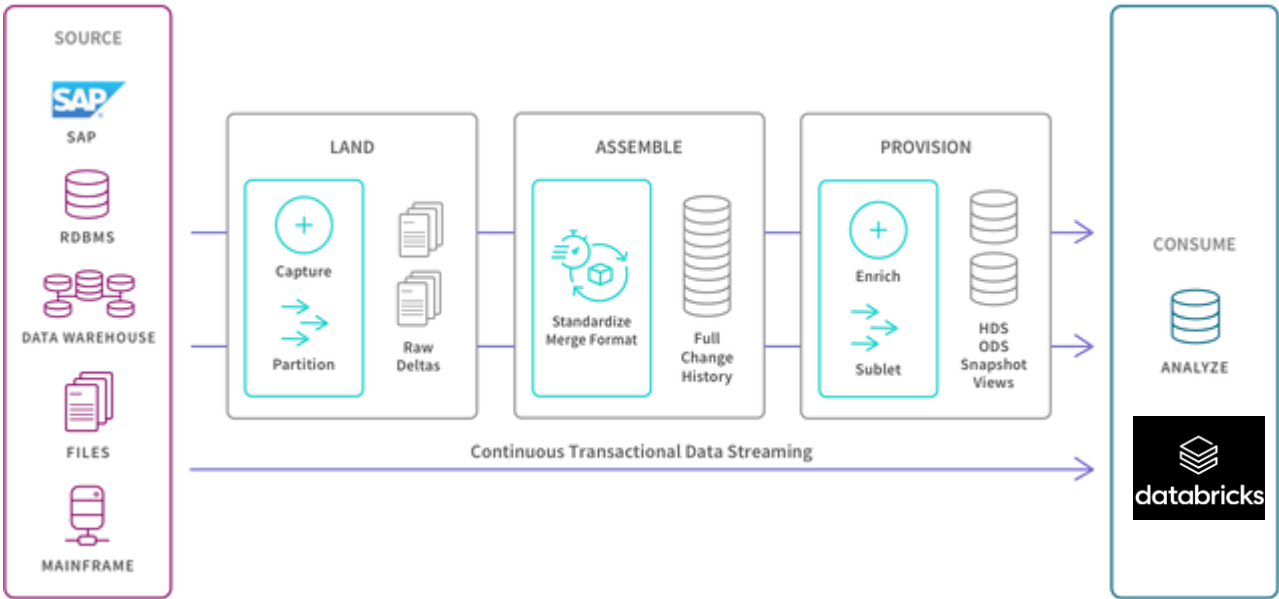
Qlik Replicate

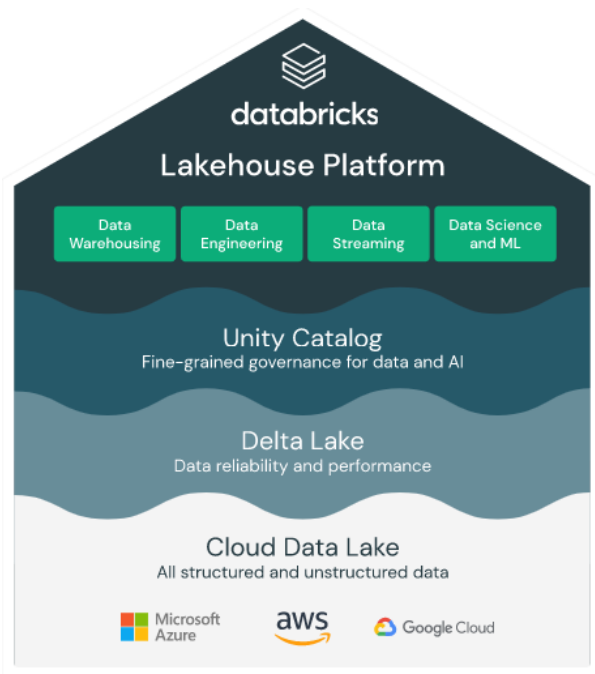
Qlik Replicate® continuously automates CDC data movement from multiple data sources (e.g., Oracle, Microsoft SQL Server, SAP, mainframe, and more) to the Databricks Lakehouse Platform. It helps the customer to avoid the heavy lifting associated with manually extracting data, transferring it via API/script, then slicing, staging, and importing it.



Qlik Compose

Qlik Compose® is a metadata driven Data Warehouse Automation and Data Lake Management tool. This solution automates the design, implementation, and updates of data warehouses, data marts and Data lakes. It minimizes the existing manual, error-prone design processes of data modeling, ETL coding, and scripting. Deploying Qlik Compose speeds up data transformation projects, helping a company to achieve greater agility, reducing risk, and realizing the potential of the Databricks Lakehouse Platform





Databricks Lakehouse Platform

The Databricks Lakehouse Platform combines the elements of data lakes and data warehouses to deliver the reliability, strong governance, and performance of data warehouses with the openness, flexibility, and machine learning support of data lakes.

This approach simplifies modern data stack by eliminating the data silos that traditionally separate and complicate data engineering, analytics, BI, data science and machine learning. It is built on open source and open standards to maximize flexibility. And its common approach to data management, security and governance helps organizations to run more efficiently and innovate faster.

Qlik Data Integration Solutions for Databricks

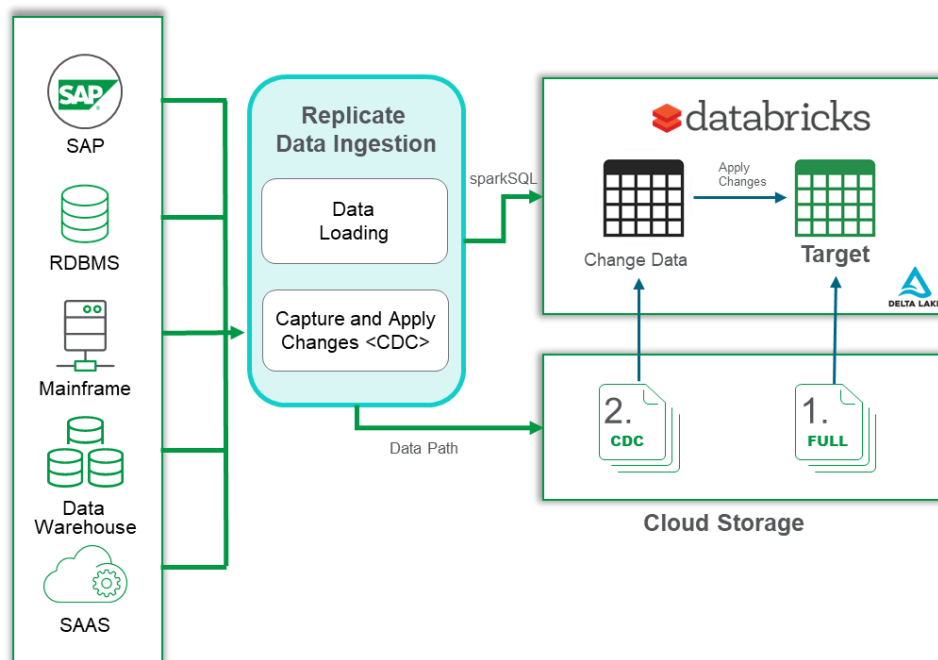
Qlik Data Integration supplies two architectural options for ingesting data to the Databricks platform. These choices supply flexibility for differing latency, compute, cost and functional requirements. Based on your Qlik Data Integration version, there are some options to get data into Databricks Lakehouse

1. Before August / 2022 version Qlik Replicate Databricks Lakehouse (Delta) Endpoint

Qlik Replicate® continuously automates CDC from multiple data sources (e.g., Oracle, Microsoft SQL Server, SAP, mainframe, and more) to the Databricks Lakehouse Platform. It eliminates the heavy lifting associated with manually extracting data, transferring it via API/script, then slicing, staging, and importing it. Qlik Replicate's Databricks Delta Endpoint (currently supported on Azure

only) supplies frictionless, real-time ingestion and automated incremental application of changes to Databricks Delta tables. In this architecture, Qlik Replicate performs the following functions:

1. Instantiate the target
 - a. Create target tables in DELTA format with proper data types translated from the source
 - b. Perform an initial / full load from the source
2. Capture and apply changes
 - a. Capture changes using log-based CDC from source
 - b. Deliver and APPLY changes (Insert / Update / Deletes) to the target DELTA tables
3. Qlik Replicate delivers data to Databricks via cloud storage with the apply process handled via Spark SQL execution over ODBC



2. Qlik Replicate + Qlik Compose for non-Delta endpoints

There are situations when a Delta endpoint cannot be used or there is a requirement that is not supported natively by Qlik Replicate:

- Current versions of Qlik Replicate AWS and GCP endpoints that do not support Delta
- Having a current and historical data requirement

- A lot of transformations and lookup values needed to be applied before loading in Databricks Delta tables
- A requirement to have a delayed MERGE process to reduce the amount of processing required against Databricks while still providing access to low latency changes

For those situations Qlik Replicate can work along with Qlik Compose to leverage a low-latency data ingestion in Databricks Delta. In this architecture:

Qlik Replicate performs the following:

1. Instantiate the target
 - a. Create target tables as external (pointing to configured cloud storage) with proper data types translated from the source
 - b. Perform an initial / full load from the source to cloud storage
2. Capture and store changes
 - a. Capture changes using log-based CDC from source
 - b. Deliver changes (Insert / Update / Deletes) to the configured cloud storage target change tables

Qlik Compose performs the following:

1. Create tables to house a current (ODS) view of data, and optionally a historical data set (HDS) which provides type 2 change tracking.
2. Process data for Delta Lake consumers through scheduled micro-batch execution of the Compose workflow
3. Create “live views” that union latest unprocessed changes in the change table with already processed data at read time

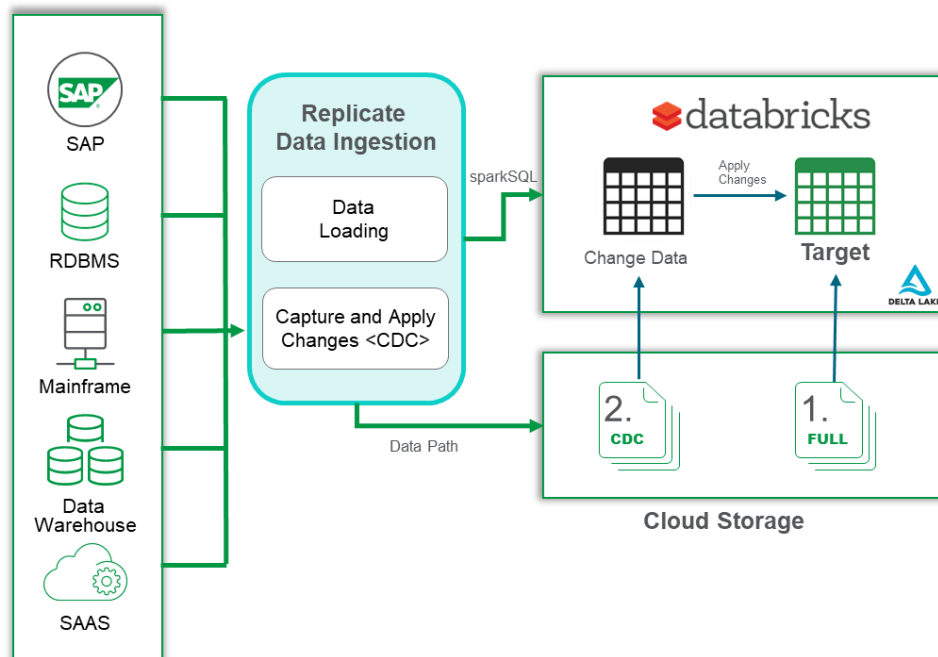
A more elaborate discussion about the two approaches and a decision guide between them is available in Qlik Community: <https://community.qlik.com/t5/Knowledge/Low-Latency-Data-Integration-for-Data-Lakes-with-Qlik-Compose/ta-p/1766349>

3. Databricks Lakehouse (Delta) Connector (Available with Qlik Replicate and Qlik Compose version August 2022 or after)

The “Databricks Lakehouse (Delta)” connector supports Databricks running on AWS, Azure and GCP. This connector works like the “Qlik Replicate Databricks Delta Endpoint” (described in section 1) and also supports “Databricks SQL Warehouse” clusters.

In this architecture, Qlik Replicate performs the following

1. Instantiate the target
 - a. Create target tables in DELTA format with proper data types translated from the source
 - b. Perform an initial / full load from the source
2. Capture and apply changes
 - a. Capture changes using log-based CDC from source
 - b. Deliver and APPLY changes (Insert / Update / Deletes) to the target DELTA tables
3. Qlik Replicate delivers data to Databricks via cloud storage with the apply process handled via Spark SQL execution over ODBC



Recommendations for Optimizing Qlik Replicate CDC with Databricks

Lakehouse (Delta) connectors

DISCLAIMER


The recommendations below are for reference and informational purposes only, and are based on projects and POCs conducted by Qlik and its partners with respect to specific requirements. Several factors like network topology, latency, table structure, update frequency, driver versions, etc. may affect the necessary settings for your particular use case. Customers are advised to perform the necessary scoping and diligence to determine their own requirements.

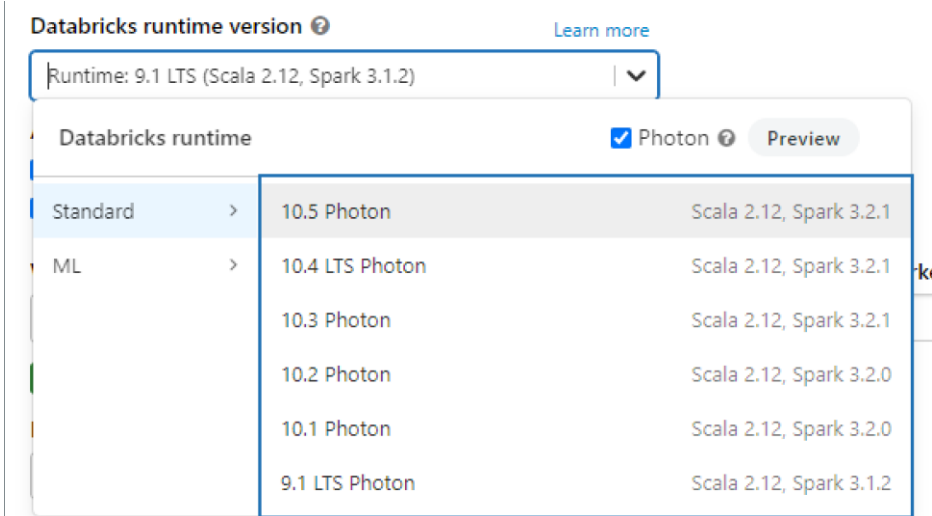
Scope : SQL Warehouses

1. Databricks runtime version supporting Photon

Scope : Interactive Clusters

If you are using a general purpose cluster as your compute cluster select “Photon” when you are selecting the **Databricks runtime version** that will support your general purpose cluster. Photon is the native vectorized query engine on Databricks, written to be directly compatible with Apache Spark. Photon is part of a high-performance runtime that runs your existing SQL and DataFrame API calls faster and reduces your total cost per workload. For a fuller discussion about Photon, please refer to this document <https://docs.databricks.com/runtime/photon.html>

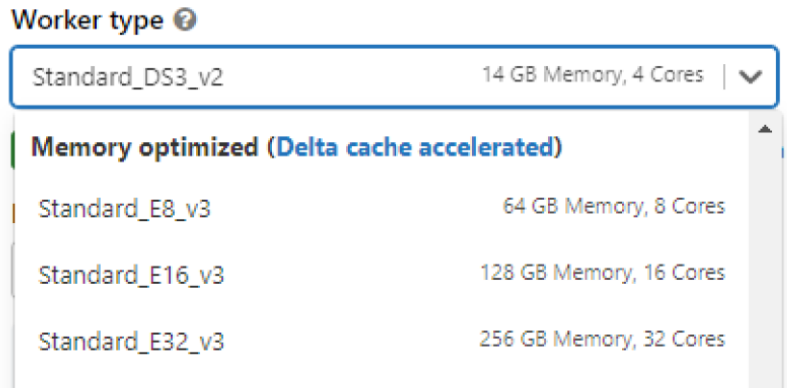
	Photon is used by default in Databricks SQL warehouses.
---	---



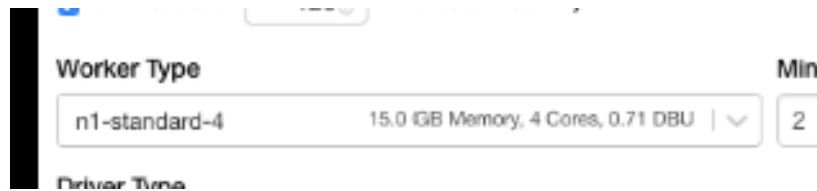
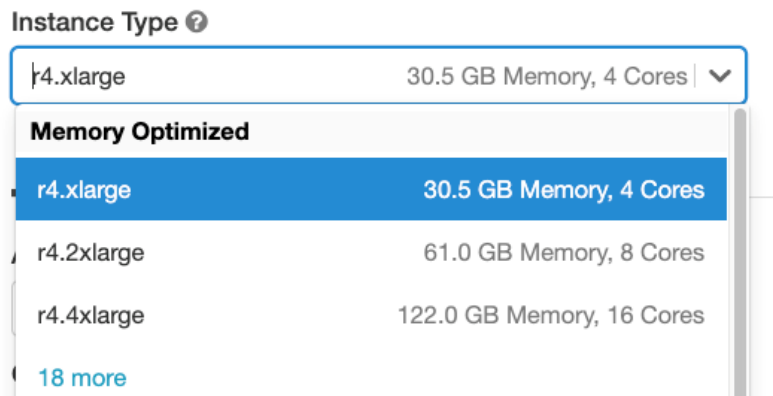
2. Select “Memory optimized – Delta cache accelerated”

Scope : Interactive Clusters

When you are configuring your cluster, make sure you select the “Memory optimized – Delta cache accelerated” **Worker type** .



(*) the list above is based on Azure Databricks, this may change if you´re using AWS or GCP



3. Configure Auto-Optimize options

Scope: Interactive Clusters

If you are using one endpoint that supports Delta tables, add a configuration to your cluster to enable `optimizeWrite` and disable `autoCompact`. Disabling `autoCompact` is necessary to prevent serial compaction from being triggered by real-time CDC updates (which can lead to increased latency). To do that, add the lines below to your **Spark** section of **Advanced options** of your cluster.

```
spark.databricks.delta.properties.defaults.autoOptimize.optimizeWrite true
spark.databricks.delta.properties.defaults.autoOptimize.autoCompact false
```

Please check <https://docs.databricks.com/clusters/configure.html> for more information about configuring your cluster.

▼ Advanced options

Azure Data Lake Storage credential passthrough ⓘ Available on Azure Databricks premium [Learn more](#)

Enable credential passthrough for user-level data access

Spark Tags Logging Init Scripts

Spark config ⓘ

```
spark.databricks.delta.properties.defaults.autoOptimize.optimizeWrite true
spark.databricks.delta.properties.defaults.autoOptimize.autoCompact false
```

4. Optimize tables on a regular basis

It is important to schedule a notebook to OPTIMIZE tables in your Delta Lake. This will improve query speed for the data landed. Please consult this documentation:

<https://docs.microsoft.com/en-us/azure/databricks/delta/optimizations/file-mgmt> for samples of notebooks to optimize the tables.

5. Table selections in a task

Scope : Interactive Clusters / SQL Warehouses

Databricks recommends isolating large or wide (many columns) tables that do a lot of processing to their own tasks. Using this approach makes it easier to monitor the performance or allocate a cluster to a specific task.

6. Auto scaling

Scope : Interactive Clusters / SQL Warehouses

Due to the variable workload volumes that CDC presents, the recommendation is to review your configuration based on the workload and testing with your tasks, monitoring and then increasing or decreasing based on the usage. Please refer to Databricks documentation (<https://docs.databricks.com/clusters/clusters-manage.html#monitor-performance>) for how to monitor cluster performance.

7. File size configuration

Scope : Interactive Clusters / SQL Warehouses

There is one Qlik Replicate parameter at the connection level that could increase the data throughput. It is called **Maximum file size(MB)** and it is located under **Advanced** settings of your connection



The default value is 100Mb and this parameter indicates the file size that is uploaded to the staging area before being loaded into a table. You can see below the impact of changing this parameter for a table with 100M records (approx. 3.8Gb data on the source). There is no “golden rule” for this parameter, but usually bigger file size increases the performance of the data transfer which is very important during the initial full load.

Cluster configuration:

Databricks Runtime Version

9.1 LTS Photon (includes Apache Spark 3.1.2, Scala 2.12)

Autopilot options

Enable autoscaling [?](#)

Terminate after minutes of inactivity [?](#)

Worker type [?](#)

Standard_E8_v3 64 GB Memory, 8 Cores

Workers Current

4

0

Spot instances [?](#)

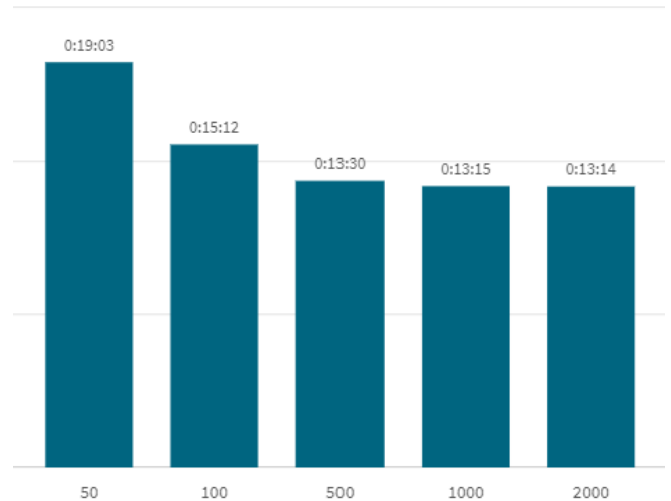
Driver type

Standard_E8_v3 64 GB Memory, 8 Cores

Source table (Azure RDS MySQL)

Category	Table Name	Storage Type	Size
Procedures	employees	InnoDB	14M
Triggers	salaries	InnoDB	86M
Events	salaries_replication	InnoDB	3.8G

Maximum file size (Mb)	Elapsed Time Full Load
50	00:19:03
100	00:15:12
500	00:13:30
1000	00:13:15
2000	00:13:14



As shown above, there was very good improvement when increasing the file size from the default value (100MB) to 500MB, though additional increases beyond 500MB in this test had much less impact on performance.

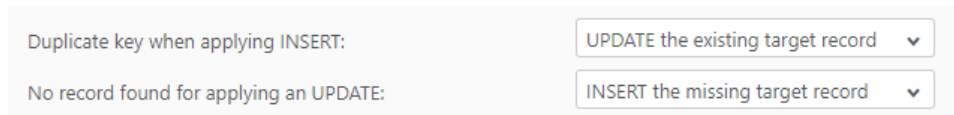
8. Batch Tuning settings

Qlik Replicate micro-batches changes for optimized delivery to Databricks Delta and batch tuning configuration for a task impacts the size of the micro-batch sent to Databricks.

- a. Change Processing Mode: Only Batch Optimized Apply is supported for Databricks targets.
- b. Apply batched changes to multiple tables concurrently: This option configures the number of threads that will work in parallel to upload and apply data to Databricks. The default value is 5, with a maximum of 50. Increasing this value can improve your throughput when there are many tables with CDC in a given batch however it may require additional cluster resources. Please review limitations of this mode in Replicates help guide.
- c. Apply batched changes in intervals settings configure the time and size of the micro-batch.
- d. **Longer than (seconds):** This specifies the minimum amount of time to wait between each application of batch changes. The default value is 1 and typically is too low a value for Databricks delta apply processes. Increasing this value decreases the frequency with which changes are applied to the target while increasing the size of the batches, essentially creating larger batches at the expense of some additional latency. It is recommended to start with a value of 60 and increase even further if some additional latency is acceptable. In some cases waiting for larger batches can actually improve throughput and latency
- e. **But less than (seconds):** This value specifies the maximum amount of time to wait between each application of batch changes (before declaring a timeout). In other words, the maximum acceptable latency. The default value is 30. This value determines the maximum amount of time to wait before applying the changes, after the **Longer than (seconds)** value has been reached. It is recommended to configure this value to 120 (combined with a Longer Than value of 60 and tuning the value even higher if more latency is acceptable).
- f. **Force apply a batch when processing memory exceeds (MB):** this setting specifies the maximum amount of memory to use for pre-processing in Batch optimized apply mode. The default value is 500. For maximum batch size, set this value to the highest amount of memory you can allocate to Qlik Replicate. It is recommended to start with a value of 2000 and consider tuning higher if there are enough resources on the Qlik Replicate server.

9. Changing from I/U/D to D/I (upsert mode in Qlik Replicate)

Qlik Replicate has multiple methods to update data in Databricks delta. The default mode is to send bulk INSERT / UPDATE / DELETE statements for each table that has CDC in a given micro-batch. In some cases, it can be more efficient to adjust the task to apply data using a Delete+Insert pair (Delete records in target where the key exists in the micro-batch and then insert the change records). To force this mode, Error Handling configuration must be set per the below image.



The image shows a configuration interface for Qlik Replicate. It features two rows of configuration options. The first row is labeled 'Duplicate key when applying INSERT:' and has a dropdown menu set to 'UPDATE the existing target record'. The second row is labeled 'No record found for applying an UPDATE:' and has a dropdown menu set to 'INSERT the missing target record'.

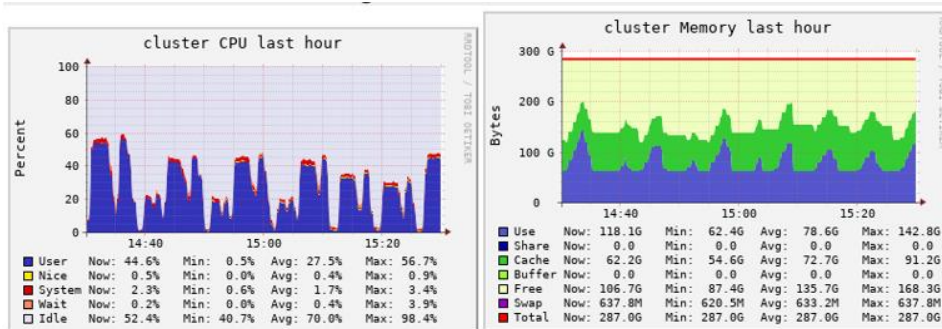
10. Partition large tables

Scope : Interactive Clusters / SQL Warehouses

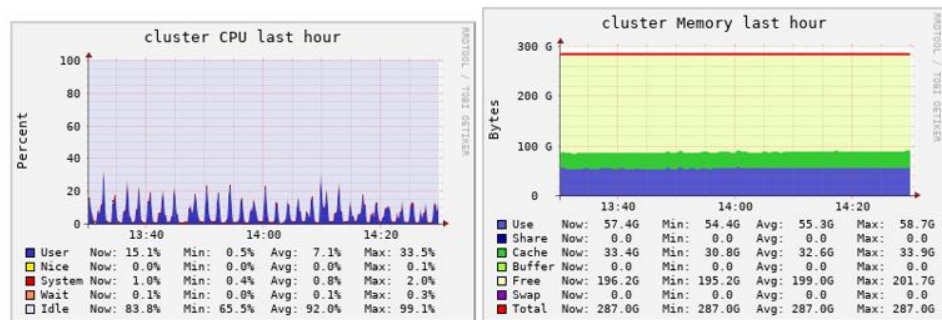
Databricks provides the ability to partition delta tables. It is recommended to partition large tables that could be a bottleneck in the apply process. Qlik Replicate does not currently support configuring target partitioning within the task. The target delta table should be created by Qlik Replicate, then re-created with the appropriate partition columns. If a table is defined as partitioned, it is recommended to set the task to perform a TRUNCATE for full loads.

While partitioning is a straightforward concept, determining the best partitioning column(s) requires a solid understanding of how data is modified by the application. It is not recommended to partition on the primary key due to cardinality concerns. Large tables that require partitioning are typically “transactional” in nature – e.g. sales data. Typically selecting a date column or adding a YEAR_MONTH column to the target data set within Replicate provides a good method for partitioning. Below is an example of partitioning impact to cluster utilization and thus latency. In this example a table of approximately 68 million source rows / 655 GB of data was processing a production CDC workload. Partitioning the delta table using a DATE column achieved a 73% reduction in latency and a large reduction in memory and CPU consumption on the cluster.

Cluster Utilization – Not Partitioned



Cluster Utilization – Partitioned

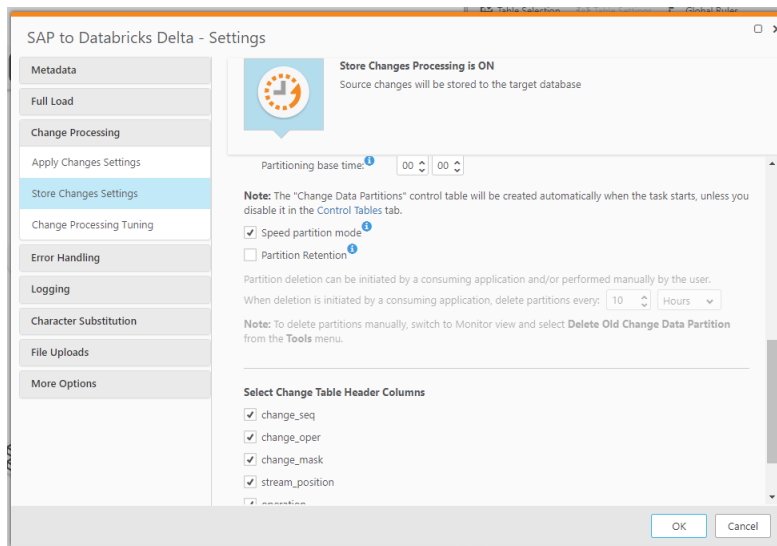


Recommendations for Optimizing Qlik Replicate and Compose Model

Recommendations from 1-10 from previous section are still valid

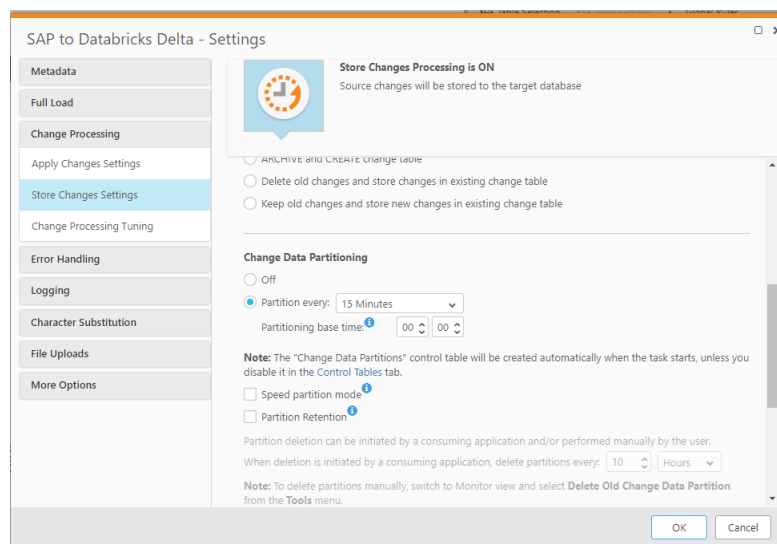
11. Enable Speed Partitions

Speed partitioning is a feature in Qlik Replicate, designed to improve the read latency of data delivered to Databricks. Qlik Replicate delivers data to Databricks using time-based partition windows. This architecture creates the partition at that start of the time window instead of at the end. This allows Qlik Compose to manage ingestion partitions, with the added benefit of being able to read data as soon as it has been delivered instead of having to wait for the time-based partition window to close. The merge-on-read perspective created by Qlik Compose is known as a Live View. This feature can be enabled at your **Task Settings** option under the **Change Processing / Store Changes Settings**



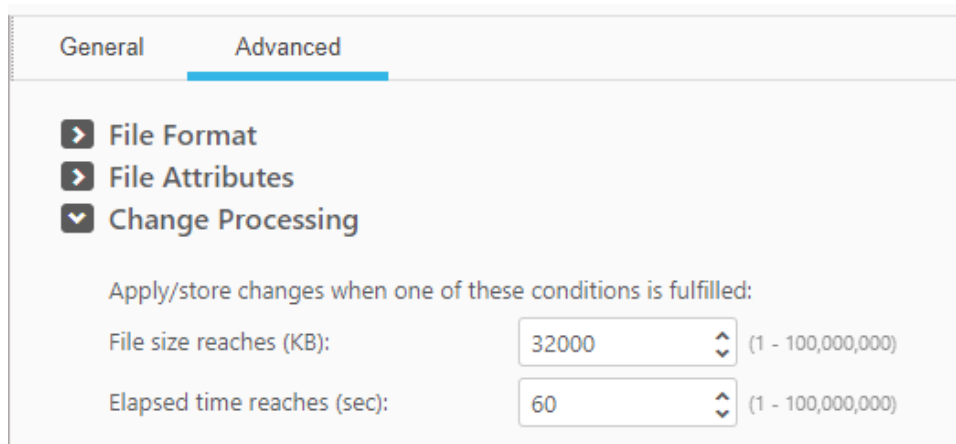
12. Partition Timeframe

To prevent partition / metadata contention on Databricks, it is recommended that Store Changes partitions have one interval of at least 15 minutes. This configuration is in your **Task Settings** option under the **Change Processing / Store Changes Settings**



13. Replicate file delivery

In the Qlik Replicate and Qlik Compose architecture, Qlik Replicate delivers files with CDC data to your cloud storage without Databricks interaction. Latency for file delivery is impacted by the Databricks endpoint advanced configuration. Under the Advanced tab, **Change Processing** section, you can configure the **File Size** and **Elapsed time** configurations. Qlik Replicate will send the CDC files as soon as one of these conditions is met.



14. Compose processing interval

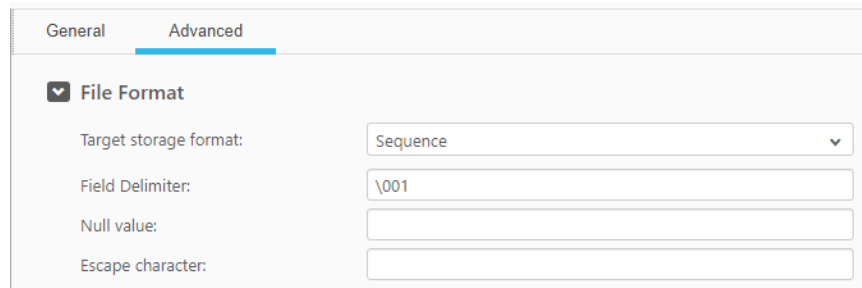
Qlik Compose processes partitions that Replicate has *closed*. A partition is *closed* when Qlik Replicate has completed delivery of CDC data for the given partition time interval (e.g. 30 minutes). Qlik Compose can be configured to run tasks in shorter intervals than Qlik Replicate partition intervals, however if there are no unprocessed closed partitions, Qlik Compose will not have any workload to perform. Note that while the Qlik Replicate change partition interval impacts the materialized data set, Qlik Compose live views provide a merge-on-read capability to access data with low latency.

15. Latency impact

The above configurations (partition timeframe and file delivery) impact Qlik Compose processing interval and live view latency. Qlik Compose processing is impacted by the partition interval. Live view latency is impacted by Qlik Replicate file delivery configuration. As soon as files are delivered to the target cloud storage, the data is accessible via Qlik Compose live views.

16. SEQUENCE as Target storage format

When using Text as the target storage format, new lines as part of the data values are not supported (due to an Apache Hadoop limitation). Best practice is to use Sequence as the target storage format. To configure that, open the **Advanced** tab of your data connection and select “Sequence” as **Target storage format**



The image shows a configuration interface with two tabs: 'General' and 'Advanced'. The 'Advanced' tab is selected. Under the 'File Format' section, there are four fields:

- Target storage format:** A dropdown menu with 'Sequence' selected.
- Field Delimiter:** A text input field containing '\001'.
- Null value:** An empty text input field.
- Escape character:** An empty text input field.

Appendix I - Creating Qlik Replicate Connections

Qlik Replicate currently only supports Databricks Lakehouse Platform as a target in the versions below:

- Databricks on AWS
- Databricks on Google Cloud
- Microsoft Azure Databricks
- Microsoft Azure Databricks Delta



It is highly recommended that you check the documentation available on help.qlik.com for the latest requirements and supported configurations for your Databricks environment

1. Obtaining Databricks credentials and connection details

To create a data connection in Qlik Replicate to a Databricks instance you will need:

1. Server Hostname
2. Port
3. HTTP Path
4. Token

• Clusters

Information 1, 2 and 3 can be extracted from Databricks console going to the cluster configuration or your SQL Endpoint configuration and under the **Advanced Options** section you will find the **JDBC/ODBC** tab

The screenshot shows the Databricks console interface for a cluster named 'replicate-interactive'. The 'JDBC/ODBC' tab is selected under the 'Advanced Options' section. The configuration fields are as follows:

- Server Hostname: dbc-bddf00fa-7593.cloud.databricks.com
- Port: 443
- Protocol: HTTPS
- HTTP Path: sql/protocolv1/o/1782382280407890/0815-183031-jzubmjs0

- Databricks SQL Warehouse
Information 1, 2 and 3 can be extracted from Databricks console going to the **SQL Warehouse** section under the “Connection details” tab

Overview **Connection details** Monitoring

Server hostname
dbc-bddf00fa-7593.cloud.databricks.com

Port
443

Protocol
https

HTTP path
/sql/1.0/warehouses/60cc3d41bc5c692b

To get an access token, you need to go to **User Settings** section of your Databricks console and use the **Generate new token** button.

User Settings

Query snippets **Personal access tokens** Account

Use personal access tokens to authenticate to the Databricks REST API. [Learn more](#)

+ Generate new token

Generate token

Comment

Lifetime (days)

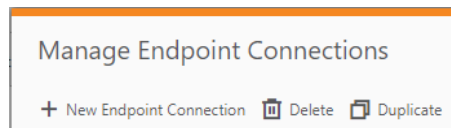
Cancel **Generate**

It is important to store the generated token in a safe place, because you cannot retrieve it again after you close this dialogue

2. Databricks on AWS

To configure a connection to a Databricks on AWS instance, follow these steps:

- 1) Carefully review the help.qlik.com documentation about “Prerequisites” and “Limitations and considerations”
- 2) Install the Simba Spark ODBC driver on the Qlik Replicate server machine. Check the documentation for required versions
- 3) On your Qlik Replicate Console, go to section **Manage Endpoint Connections**



- 4) Click on **New Endpoint Connection**
- 5) Give your connection a meaningful name and a description
- 6) Select **Target** as the **Role** and Databricks on AWS as **Type**

- 7) For the **Databricks ODBC Access** section you need the information you collected using the section *“Obtaining Databricks credentials and connection details”*
 1. **Host** - The host name of the Databricks workspace
 2. **Port** - The port via which to access the workspace
 3. **Token** - Your personal token for accessing the workspace.
 4. **HTTP Path** - The path to the cluster.
 5. **Database** - The name of the Databricks target database. You can browse for existing databases. You can also check Databricks documentation for how to create a new database.
- 8) For the **Amazon S3 Storage Data Access**, first choose which method your Databricks cluster uses to access the Amazon S3 storage: **Access Directly** (the default) or **Access through DBFS Mount**. Direct Access requires configuring the cluster with right spark configurations to access the S3 bucket.



Direct access is the recommended practice as it is more secure than mounting your cloud storage. Unless you have a specific requirement, use this option

Then fill the form using the required credentials for a previously created S3 bucket

Replicate S3 Storage Access:

Bucket name:

Bucket region: ▼

Access type: ▼

Access key:

Secret key:

Target folder:

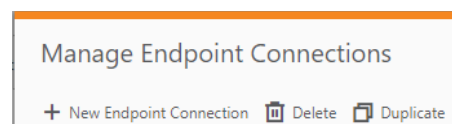


- If you have more than one connection to the same Databricks instance, it is recommended that you use the same Storage Data Access option for all connections. It is possible to have different connections with different options- check Qlik Replicate Help (https://help.qlik.com/en-US/replicate/May2022/Content/Replicate/Main/Databricks/databricks_set_up_hadoop_as_target.htm) to get the procedure to use this approach
- Note that this endpoint supports only Full Load and Storage Changes configuration. Apply changes is only supported in "Delta" endpoints.

3. Databricks on Google Cloud

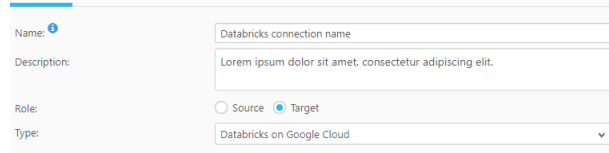
To configure a connection to a Databricks on Google Cloud instance, follow these steps:

- 1) Carefully review the help.qlik.com documentation about “Prerequisites” and “Limitations and considerations”
- 2) Install the Simba Spark ODBC driver on the Qlik Replicate server machine. Check the documentation for required versions
- 3) On your Qlik Replicate Console, go to section **Manage Endpoint Connections**
- 4) Click on **New Endpoint Connection**



- 5) Give your connection a meaningful name and a description

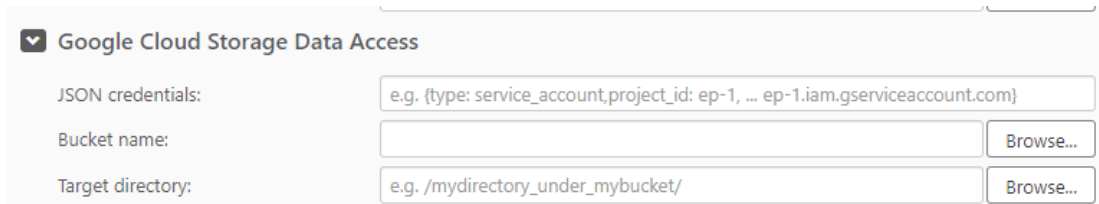
6) Select **Target** as the **Role** and Databricks on Google Cloud as **Type**



7) For the **Databricks ODBC Access** section you need the information you collected using the section “*Obtaining Databricks credentials and connection details*”

- a. **Host** - The host name of the Databricks workspace
- b. **Port** - The port via which to access the workspace
- c. **Token** - Your personal token for accessing the workspace.
- d. **HTTP Path** - The path to the cluster being used.
- e. **Database** - The name of the Databricks target database. You can browse for existing databases. You can also check Databricks documentation for how to create a new database.

8) For the **Google Cloud Storage Data Access**, you need to supply your credentials in JSON format and the bucket name and target directory from your previously created bucket



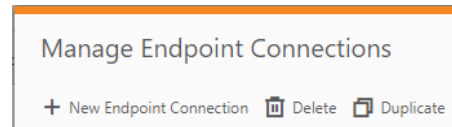
Note that this endpoint supports only Full Load and Storage Changes configuration. Apply changes is only supported in "Delta" endpoints.

4. Microsoft Azure Databricks

To configure a connection to a Microsoft Azure Databricks instance, follow these steps:

- 1) Carefully review the help.qlik.com documentation about “Prerequisites” and “Limitations and considerations”
- 2) Install the Simba Spark ODBC driver on the Qlik Replicate Server machine. Check the documentation for required versions
- 3) On your Qlik Replicate Console, go to section **Manage Endpoint Connections**

4) Click on **New Endpoint Connection**



5) Give your connection a meaningful name and a description

6) Select **Target** as the **Role** and Microsoft Azure Databricks as **Type**

7) For the **Databricks ODBC Access** section you need the information you collected using the section “*Obtaining Databricks credentials and connection details*”

- a. **Host** - The host name of the Databricks workspace
- b. **Port** - The port via which to access the workspace
- c. **Token** - Your personal token for accessing the workspace.
- d. **HTTP Path** - The path to the cluster being used.
- e. **Database** - The name of the Databricks target database. You can browse for existing databases. You can also check Databricks documentation for how to create a new database.

8) For the **Azure Storage Data Access**, choose which method your Databricks cluster uses to access the ADLS storage: **Access Directly** (the default) or **Access through DBFS Mount**. Direct Access requires configuring the cluster with right spark configurations to access the ADLS bucket.



Direct access is the recommended practice as it is more secure than mounting your cloud storage. Unless you have a specific requirement use this option

Then fill the form using the required credentials for a previously created ADLS bucket

▼ Azure Storage Data Access

Databricks storage access method: Access Directly
 Access through DBFS Mount

Replicate Cloud Storage Access:

Storage type:

Storage account:

Azure Active Directory ID:

Azure Active Directory application ID:

Azure Active Directory application key:

Container name:

Target folder:

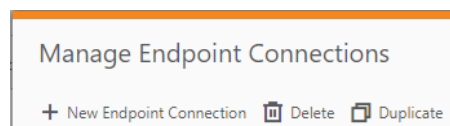


Note that this endpoint supports only Full Load and Storage Changes configuration. Apply changes is only supported in "Delta" endpoints.

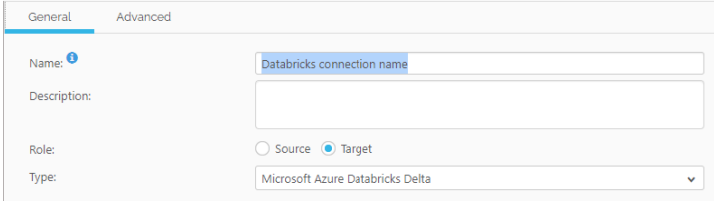
5. Microsoft Azure Databricks Delta

To configure a connection to a Microsoft Azure Databricks Delta instance, follow these steps:

- 1) Carefully the help.qlik.com documentation about “Prerequisites” and “Limitations and considerations”
- 2) Install the Simba Spark ODBC driver on the Qlik Replicate server machine. Check the documentation for required versions
- 3) On your Qlik Replicate Console, go to section **Manage Endpoint Connections**
- 4) Click on **New Endpoint Connection**



- 5) Give your connection a meaningful name and a description
- 6) Select **Target** as the **Role** and Microsoft Azure Databricks Delta as **Type**



The screenshot shows a configuration window with two tabs: 'General' and 'Advanced'. The 'General' tab is active. It contains the following fields:

- Name:** A text input field containing 'Databricks connection name'.
- Description:** An empty text input field.
- Role:** Radio buttons for 'Source' and 'Target'. The 'Target' radio button is selected.
- Type:** A dropdown menu showing 'Microsoft Azure Databricks Delta'.

- 7) For the **Databricks ODBC Access** section you need the information you collected using the section “*Obtaining Databricks credentials and connection details*”
 - a. **Host** - The host name of the Databricks workspace
 - b. **Port** - The port via which to access the workspace
 - c. **Token** - Your personal token for accessing the workspace.
 - d. **HTTP Path** - The path to the cluster being used.
 - e. **Database** - The name of the Databricks target database. You can browse for existing databases. You can also check Databricks documentation for how to create a new database.
- 8) For the **Azure Storage Data Access**, you need choose which method your Databricks cluster uses to access the ADLS storage: **Access Directly** (the default) or **Access through DBFS Mount**. Direct Access requires configuring the cluster with right spark configurations to access the ADLS bucket. This is the recommended practice as it is more secure than mounting your cloud storage. Mounting cloud storage is a viable method to integrate with storage accounts, however there are no access controls in place and DBFS mounts are accessible from all clusters within a Databricks workspace. Qlik Replicate needs to know those configuration options so that creation of the EXTERNAL tables is performed with correct location syntax. You should refer to Databricks documentation to decide which choice to follow. Recommended links:
 - <https://docs.databricks.com/data/data-sources/azure/adls-gen2/index.html>
 - <https://databricks.com/blog/2017/08/23/best-practices-for-coarse-grained-data-security-in-databricks.html>

Then you need to configure the endpoint for your staging storage

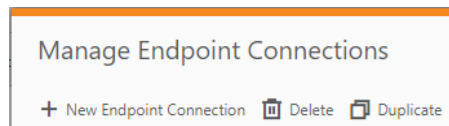
The screenshot shows a configuration form for a staging endpoint. It is titled "Staging" with a dropdown arrow. The form contains the following fields:

- Storage type: Azure Data Lake Storage (ADLS) Gen2 (dropdown)
- Storage account: e.g. myadls (text input)
- Azure Active Directory ID: e.g. 12854717-3c57-4816-ab29-0c418c8310a1 (text input)
- Azure Active Directory application ID: e.g. 02d3410d-a45c-46e0-89aa-b98dd3f25b03 (text input)
- Azure Active Directory application key: e.g. MhccSGMofGshE7xvN+y+e9qVxEUfOgbUi8HFB1GqRbs= (text input)
- File system: e.g. myfilesystem (text input)
- Staging directory: e.g. /myfilesystem/mydirectory (text input) with a "Browse..." button to its right.

6. Databricks Lakehouse (Delta)

To configure a connection to a Databricks Lakehouse (Delta) instance, follow these steps:

- 1) Carefully the help.qlik.com documentation about “Prerequisites” and “Limitations and considerations”
- 2) Install the Simba Spark ODBC driver on the Qlik Replicate server machine. Check the documentation for required versions
- 3) On your Qlik Replicate Console, go to section **Manage Endpoint Connections**
- 4) Click on **New Endpoint Connection**



- 5) Give your connection a meaningful name and a description
- 6) Select **Target** as the **Role** and Databricks Lakehouse (Delta) as **Type**

The screenshot shows the configuration form for a new endpoint connection. It includes the following fields:

- Name: New Endpoint Connection 1 (text input)
- Description: (empty text input)
- Role: Target (radio button selected, Source is unselected)
- Type: Databricks Lakehouse (Delta) (dropdown menu)

- 7) For the **Databricks ODBC Access** section you need the information you collected using the section “*Obtaining Databricks credentials and connection details*”
- Host** - The host name of the Databricks workspace
 - Port** - The port via which to access the workspace
 - Token** - Your personal token for accessing the workspace.
 - HTTP Path** - The path to the cluster being used.
 - Database** - The name of the Databricks target database. You can browse for existing databases. You can also check Databricks documentation for how to create a new database.
- 8) For the **Staging** section you need choose which storage type will be used to load intermediate data. There are three options
- Amazon S3
 - Google Cloud Storage
 - Microsoft Azure Data Lake Storage (ADLS) Gen2



Even though it is possible to have your Databricks instance running on a particular hyperscaler and your storage zone being registered from another it is highly recommended using the storage from your Databricks instance

For **Amazon S3**, provide the following information:

- In the **Storage type** field, select **Amazon S3**.
- In the **Bucket name** field, specify the name of your Amazon S3 bucket.
- In the **Bucket region** field, specify the region where your bucket is located. It is recommended to leave the default (Auto-Detect) as it usually eliminates the need to select a specific region. However, due to security considerations, for some regions (for example, AWS GovCloud) you might need to explicitly specify the region. To do this, select Other and specify the code in the Region code field.

For a list of region codes, see the Region availability section in:

<https://docs.aws.amazon.com/general/latest/gr/s3.html>

4. In the **Access options** field, choose one of the following:
 - **Key pair** Choose this method to authenticate with your Access Key and Secret Key.
 - **IAM Roles for EC2** Choose this method if the machine on which Qlik Replicate is installed is configured to authenticate itself using an IAM role.
5. In the **Access key** field, specify the access key information for Amazon S3.
6. In the **Secret key** field, specify the secret key information for Amazon S3
7. In the **Staging directory** field, specify where to create the data files on Amazon S3.

For **Google Cloud Storage**, provide the following information:

1. In the **Storage type** field, select **Google Cloud Storage**.
2. In the **JSON credentials** field, specify the JSON credentials for the service account key used to access the Google Cloud Storage bucket.
3. In the **Bucket name** field, specify the name of the bucket in Google Cloud Storage where you want the data files to be written. This must be the same as the bucket you configured for your Databricks cluster.
4. In the **Staging directory** field, specify where to create the data files in the specified bucket.

For **Microsoft Azure Data Lake Storage (ADLS) Gen2**, provide the following information:

1. In the **Storage type** field, select **Microsoft Azure Data Lake Storage (ADLS) Gen2**.
2. In the **Storage account** field, specify the name of your storage account.
3. In the **Azure Active Directory Tenant ID** field, specify the Azure Active Directory Tenant ID.
4. In the **Application Registration Client ID** field, specify the Application Registration Client ID.
5. In the **Application Registration Secret** field, specify the Application Registration Secret ID.
6. In the **Container** field, specify the name of your container.
7. In the **Staging directory** field, specify where to create the data files on ADLS.



About Qlik

Qlik's vision is a data-literate world, where everyone can use data and analytics to improve decision-making and solve their most challenging problems. Our cloud-based Qlik Active Intelligence Platform® delivers end-to-end, real-time data integration and analytics cloud solutions to close the gaps between data, insights and action. By transforming data into Active Intelligence, businesses can drive better decisions, improve revenue and profitability, and optimize customer relationships. Qlik does business in more than 100 countries and serves over 38,000 active customers around the world.

[qlik.com](https://www.qlik.com)

© 2023 QlikTech International AB. All rights reserved. All company and/or product names may be trade names, trademarks and/or registered trademarks of the respective owners with which they are associated.