



Data Catalyst®

## **Qlik Data Catalyst Specification**

Integration Setup Between Qlik Sense and Qlik Data Catalyst:

- I. [Publish to Qlik](#)
- II. [QVD Import](#)

**Release 4.4 December 2019**

**Release Date: December 23, 2019**

**Updated: January 21, 2020**

# I. Publish to Qlik

## Overview: Publish to Qlik

The following instructions detail the setup for Qlik Data Catalyst (server-side) to push data to Qlik Sense (webapp client-side) through the launch of node.js script. This integration facilitates an event-driven asynchronous runtime loop.

### 1. Create a Connector in Qlik Sense

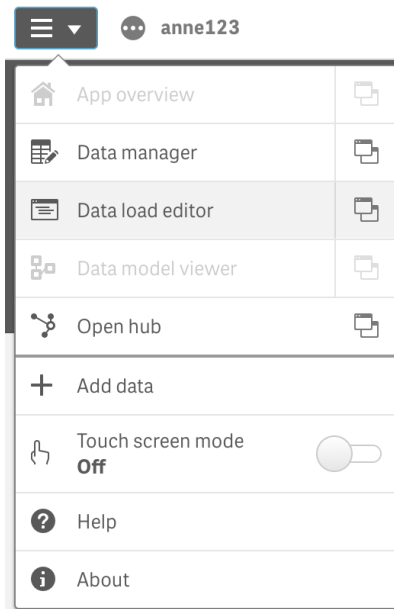
#### 1.1 Single Node Deployments: Create A Connector to PostgreSQL

When publishing to Qlik Sense from Data Catalyst running in a Single Node configuration (versus multi-node/Hadoop deployment) users should create a **connector** to the PostgreSQL distribution tables that hold views of the entities and data in Data Catalyst.

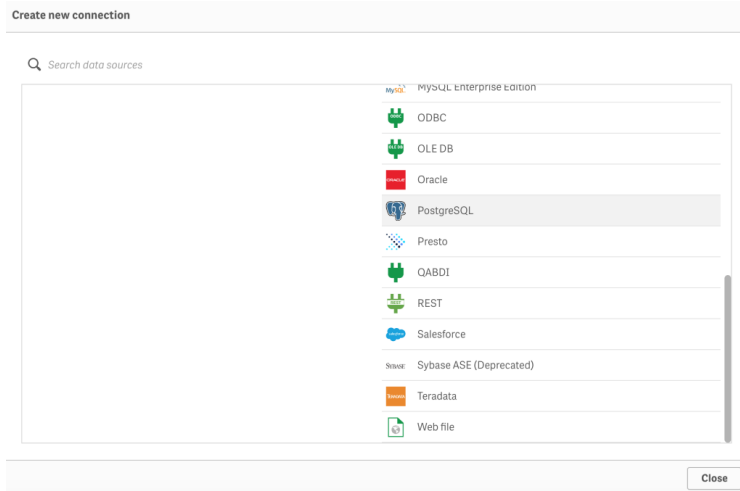
The following steps demonstrate creation of a PostgreSQL connector. (These steps are consistent with the creation of any type of Qlik Sense connector).

More information about creating PostgreSQL connectors in Qlik Sense can be found [here](#).

- 1) Log into Qlik Sense.
- 2) Select **Create New App** > enter **Name Of My App** > click **Create**
- 3) **App Overview** opens. Select **Data Load Editor** from upper-left drop-down



- 4) Select 'Create new connection' from 'Data connections' sidebar on right side of screen. Select 'PostgreSQL' data source from popup displaying available data source types.



- 5) Complete the following fields in the PostgreSQL Connection dialog box that opens:

### Database Properties

Hostname: Enter the host name of the Data Catalyst Server

Port: Enter the TCP port that PostgreSQL is listening on (defined in postgresql.conf)

Database: podium\_dist

### Authentication Information

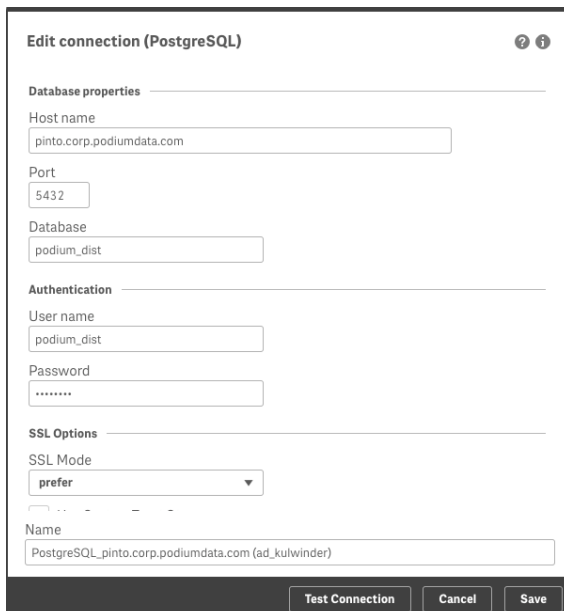
User name: enter the PostgreSQL user that will be used to authenticate the connection

Password: enter the password associated with the PostgreSQL user

### SSL Options

SSL Mode: *prefer*

- 6) **Test** the connection.  
7) **Save** the connection. Once created the connector populates in the right sidebar.



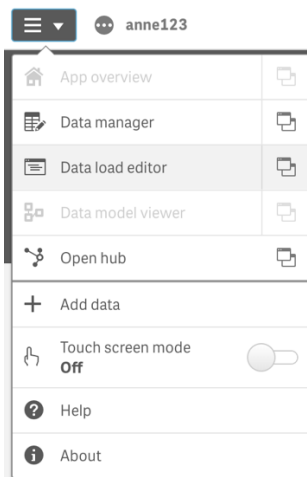
## 1.2 Multi-Node Deployments: Create A Connector to Hive

When publishing to Qlik Sense from Data Catalyst running in a Multi-Node Hadoop environment, users should create a **connector** to the **Hive** distribution tables that hold views of the entities and data in Data Catalyst.

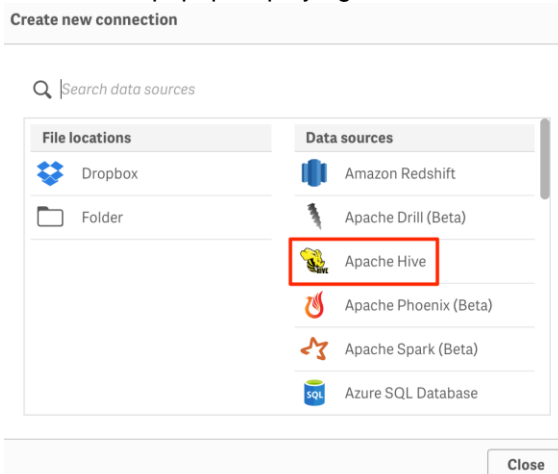
The following steps demonstrate creation of a Hive connector. (These steps are consistent with the creation of any type of Qlik Sense connector).

More information about creating Hive connectors in Qlik Sense can be found [here](#).

- 1) Log into Qlik Sense.
- 2) Select **Create New App** > enter **Name Of My App** > click **Create**
- 3) **App Overview** opens. Select **Data Load Editor** from upper-left drop-down



- 4) Select 'Create new connection' from 'Data connections' sidebar on right side of screen. Select 'Hive' data source from popup displaying available data source types.



- 5) Complete the following fields in the PostgreSQL Connection dialog box that opens:

### Database Properties

Hostname: Enter the host name of the server running the Hive Server for the Hadoop cluster

Port: Enter the TCP port that Hive is listening on (typically 10000)

Database: user\_views

## Authentication Information

**Mechanism:** Single Sign-On

**Kerberos FQDN:** Enter the fully-qualified domain name of the Kerberos authentication server. (In Active Directory environments this would be a domain controller).

**Kerberos Realm:** Enter the Kerberos realm associated with the Hadoop cluster

**KrbServiceName:** Enter the cluster Kerberos service principal being used by Hive

**Name:** Enter a name for the Hive connector

Create new connection (Apache Hive)

Database properties

Host name  
hive-server.qdc.qlik.com

Port  
10000

Database  
user\_views

Authentication

Mechanism  
Single Sign-On (Beta)

Kerberos FQDN  
kerberos1.qdc.qlik.com

Kerberos Realm  
QDC.REALM

KrbServiceName  
hive/hive-server.qdc.qlik.com@QDC.REALM

Name  
QDC-Hive-Connector

Test Connection Cancel Create

6) **Test** the connection.

7) **Save** the connection. Once created the connector populates in the right sidebar.

## 2. Copy Certificates from Qlik Sense Server to Qlik Data Catalyst node

1) Export certificates from Qlik Sense using the Management Console (QMC).

To export the public and private keys from a Qlik Sense server, please see [https://help.qlik.com/en-US/sense/June2019/Subsystems/ManagementConsole/Content/Sense\\_QMC/export-certificates.htm](https://help.qlik.com/en-US/sense/June2019/Subsystems/ManagementConsole/Content/Sense_QMC/export-certificates.htm)

The default location for the exported certificates is the following directory **on the Qlik Sense server**:

C:\ProgramData\Qlik\Sense\Repository\Exported Certificates

(A folder will be created within the “Exported Certificates” directory with whatever name is entered during the export process described in the link above).

2) Copy the certificates exported in step1 from the Qlik Sense server into the “**certs**” directory on the QDC server. The **certs** directory will be located in the QDC\_HOME directory. (QDC\_HOME was defined during the QDC installation process. The installation default is /usr/local/qdc/certs). The following files should be copied:

client.pem

client\_key.pem

```
root.pem
server.pem
server_key.pem
```

### 3. Configuration of *core\_env.properties* file on Qlik Data Catalyst machine

Set the following properties in the *core\_env.properties* file. The *core\_env.properties* is located in the TOMCAT\_HOME/conf directory. (TOMCAT\_HOME was defined during the QDC installation process. The installation default is /usr/local/apache-tomcat-<version-number>/certs).

- 1) **is.publish.to.qlik.enabled:** Set this property to **true** to display the 'Publish to Qlik' option in the Data Catalyst UI cart checkout:

```
is.publish.to.qlik.enabled=true
```

(Note that the user must logout and login to see the button after *core\_env* refresh.)

- 2) **qlik.sense.active.directory.name:** Name of the Active Directory Domain to which the Qlik Sense server belongs. This value should be enclosed by single quotes.

```
qlik.sense.active.directory.name='DOMAIN-NAME'
```

Example:

```
qlik.sense.active.directory.name='QLIKTECH'
```

- 3) **qlik.sense.url:** URL to Qlik Sense Server. It should be defined using the following format:

```
qlik.sense.url=https://<qliksense-ip-or-host-name>/sense/app/<podium-gen-app-id>
```

Examples:

```
qlik.sense.url=https://10.111.2.163/sense/app/<podium-gen-app-id>
```

```
qlik.sense.url=https://sense01.dev.qlik.com/sense/app/<podium-gen-app-id>
```

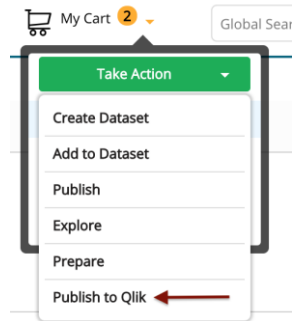
- 4) **qlik.sense.jwt:** The JSON Web Token that is used to authenticate to the Qlik Sense server.

Instructions for generating a JSON Web Token are [available here](#).

Example:

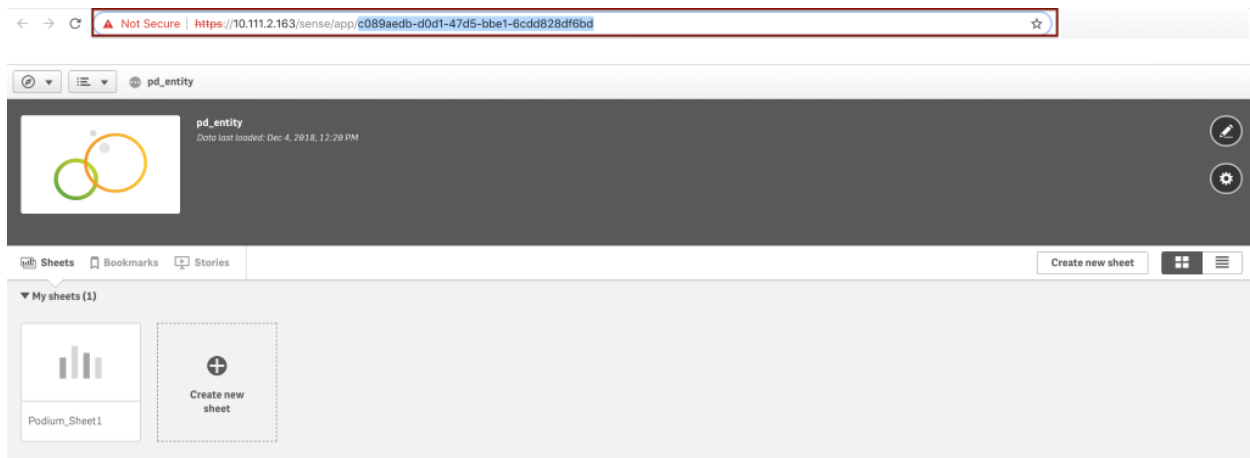
```
qlik.sense.jwt=#0{gtErmgzvXnDZpq7ooq25jzQ11HqNqjHrMlvbbv3uaef2Jftdn92rap9wLQbPLd27z54E8yuERvKC8jXHsgHeP6KeOtDg5lwkwHZr87Oqz9je8iLvwhUSHKW4VuJIWewPZFlzSk+0FeYqtp8EAaQXkksBtR3fh19Nq/qz3T2AmBvpdklzoZDnKF02pCfrvkogc1Qlb1pn7f9Jnd61G8TXPQ90H6eGBok02VtvIYj0ihljBff1f1w8VZr3ITvtdiAHujYNNbA/SYRArywz/3peasvwa1rT8ytGp6pP6YnhEtASO6iAN9gJhjV7kq1QxiWImwxuSY9YwiYjw/S8knqXeONoPHI9/OCu8DyB6+4WzM6EG6si4moLpQQAmqdamv1NfW6mlHVr+knKmtxK5kAD4XwVfbDIGY3FX6qnwrw0akVxIM31wZX57Uuscc1gK5Uut5HB4vuftvcUe5Zkytt9aw73Xi+ulPpz/iBicwfOy2OMRYbbOsPMYH8hQedag1VDwGzcb/cUjrSIXfsy6iHdTqM4D3i5TSBuhmMs1V6+dt/G/V/4yr3tm4ibwavHYQ1V+9WQnebv8CwJcGGeU/IUg==}
```

#### 4. Publish To Qlik option will now appear in UI cart checkouts:



Note: Data Catalyst generates the <podium-gen-app-id> value used in the URL when Publish To Qlik is invoked.

This value becomes the application (or report) id/url in Qlik Sense



#### 5. Confirm that the user executing the Publish To Qlik operation exists in both Qlik Data Catalyst and Qlik Sense applications.

**Congratulations! Publish to Qlik is now configured.**

## II. QVD Import

### Overview: QVD Import

Qlik Data Catalyst allows users to import QVDs (Qlik View Data) files from a mapped instance of Qlik Sense. Qlik APIs are used to pull metadata directly from Qlik Sense into Data Catalyst, and QVDs are then ingested and cataloged as source type QVD.

The process of **QVD Ingest** requires Administrators to provide information about Qlik Sense servers in the Qlik Data Catalyst application (Admin→QVD Import). Once configured, Data Catalyst queries the Sense server to obtain a list of connections and corresponding paths that are of interest and have been tagged "[QVD Catalog](#)." Data Catalyst dedupes the list and constructs a list of unique paths, and a QDC Admin must then provide a unique source name for each of those paths. Data Catalyst stores this mapping between folders and source names in its metadata for use in entity creation.

This document details environment setup in both Qlik Sense and Qlik Data Catalyst.

1. [Prerequisites + Environment Setup for QVD Import](#)
2. [Qlik Data Catalyst + Qlik Sense Relationships](#)
3. [Security](#)
4. [Mounting Qlik Sense Host Windows Directory Share on Qlik Data Catalyst Linux Host](#)
5. [Qlik Sense Virtual Proxy Setup](#)
6. [Qlik Data Catalyst QVD Import Workflow](#)
7. [Appendix](#)
  - A. [Configuring Data Catalyst Accept A Certificate Issued by an Internal Certificate Authority](#)
  - B. [Importing a Certificate Authority \(CA\) root certificate into the JVM Trust Store](#)

### Prerequisites + Environment Setup for QVD Import

**Prerequisites:** Follow the Qlik Data Catalyst Installation Guide Section 3: Installation Prerequisites. In particular, section 3.5 on Qlik Sense Integration provides details on how to install Node.js and Docker:

- **Docker:** Qlik Core is a core set of components (that includes a utility enabling conversion of proprietary Qlik format into CSV/Text Tab Delimited) that runs in a Docker container.
- **Node.js:** Used to execute Qlik Core JavaScript APIs to allow data and metadata to flow between Qlik Sense and Qlik Data Catalyst. This must be installed on both application servers (Qlik Sense and Qlik Data Catalyst).

### Environment Setup

2. The **Qlik Data Catalyst Installer** configures and enables Qlik Core and automatically populates the `core_env` property `qvds.openconnector.script.path`. This property provides the file path to a key shell script responsible for loading data. The script talks to Qlik Core (running as a Docker container) and puts the data formatted as CSV into QDC Loading Dock directory. The file can then be read like any other source in Qlik Data Catalyst.



3. **Active Directory Sync:** The **same** users and groups must be present in Qlik Data Catalyst and Qlik Sense under the same Active Directory. Qlik Sense users will either be created manually in Qlik Data Catalyst and Qlik Sense, or will be synchronized with the same Active Directory that is in use by Qlik Sense. While it is possible to manually check that the same users and groups exist in both applications, AD Sync is strongly encouraged as the preferred mechanism to ensure all users and groups are available in both applications.
  
4. **Linux Mount Point of Qlik Sense Data Share:** Qlik Sense Data Connection Windows folders must be shared, and then mounted on the Qlik Data Catalyst Linux server. See [Windows Network Share Creation + Mounting Windows Directory](#)
  
5. **Qlik Sense Virtual Proxy Setup:** The virtual proxy server is a critical gateway linking the two applications. It handles several different settings like authentication and session handling while protecting the privacy of both applications. This setup entails the creation of a JSON Web Token (JWT), which is a token generated on the Qlik Sense Server.

See [Proxy Setup](#) section.

6. **Qlik Sense configuration:** From the Qlik Management Console (QMC) in Qlik Sense, Administrators tag connectors that contain QVDs of interest with the "[QVD Catalog](#)" tag.
  
7. **QVD Ingest:** From the **QVD Ingest** tab in the Admin section of the Qlik Data Catalyst, Administrators add connectors (entry points into Qlik Sense) by providing:
  - Qlik Sense Connector details and a JSON Web Token (JWT) to authenticate to Qlik Sense
  - Directory paths to folders – this is a mapping of a Qlik Sense Windows path to a Qlik Data Catalyst Linux path

Paths can be listed from "Show QVD Paths" to see which paths have QVDs that have been "Added", "Removed" or "Changed". Users accept those statuses and the connectors are then updated accordingly. Those QVDs populate into the Source module, where data can be loaded like any other source type in Qlik Data Catalyst. See [QVD Ingest Workflow](#) section for detailed steps. Note that *Metadata* load is incremental loadtype and QVD *data* load is snapshot.

## Qlik Data Catalyst + Qlik Sense Relationships

The following table describes objects in Qlik Data Catalyst and corresponding objects in Qlik Sense.

Qlik Sense	Qlik Data Catalyst	Comments
User	User	Every Qlik Sense user must be the same user in Qlik Data Catalyst with the same name. Syncing through shared Active Directory domains is strongly encouraged. Users should have access to the same QVDs between the two

		applications. Qlik Sense is the master application where access to QVDs is defined as part of QVD authoring/administration and Qlik Data Catalyst honors these privileges. Single Sign On is in place and users shouldn't need to log into either application more than once.
<b>QVD file</b>	<b>Entity</b>	Every Qlik Sense QVD will be represented as one Entity in Qlik Data Catalyst
<b>Qlik Sense Connection</b>	<b>Group</b>	Each Qlik Sense Connection will have a corresponding group in Qlik Data Catalyst. This mapping is done for security purposes and access control management. Note that Security Groups are automatically generated, named, and synced by capturing the Qlik Sense Connector Globally Unique ID which is 36 characters and removing the hyphens to comply with Linux Group name 32-character limit.
<b>Folder</b>	<b>Source</b>	Each unique folder in Qlik Sense will be represented as one unique Source in Qlik Data Catalyst containing all Entities that represent QVDs under that folder. User access to QVDs will be governed by user access privileges as defined in Qlik Sense (via folder access).

## Security

User access to QVDs will be governed by user access privileges as defined in Qlik Sense (via folder access). The logged in user is able to access and sync QVDs for Qlik Sense connections that the user has access to and, when ingested, a Qlik Data Catalyst group (and name) will be auto-retrieved capturing the Qlik Sense Connection GUID (Globally Unique Identifier).

The folders are mapped between applications and, when the user signs in, their access to security connections in Qlik Sense are transferred to the security groups in Qlik Data Catalyst.

## Mounting Qlik Sense Windows Directory on Qlik Data Catalyst Linux Host

Each Qlik Sense Data Connection Windows folder that will be catalogued by Qlik Data Catalyst must be accessible via a Linux mount point created on the Qlik Data Catalyst server.

For example, if all Data Connection folders and QVDs are contained under C:\data on the Qlik Sense Windows server this folder should be shared and mounted as a directory on the Data Catalyst linux file system (e.g. /mnt/qvd-data). If there are multiple different Windows folders to share, then multiple Linux mounts may be needed.

## Part 1: Create a Windows Network Share on the Qlik Sense server

1. Create service account to be used by credentials file to access the Windows share.
2. Create Windows share for the location of the QVD files on the Qlik Sense server.
3. Grant the service account READ-only access to the share.

## Part 2: Mount the Qlik Sense Windows Share on the QDC Linux Server

1. Open SMB port 445 to the QDC node for the Windows share to be accessible. Note: 445 should only be opened to the QDC node. Refer to Qlik Sense documentation for more information regarding ports: [https://help.qlik.com/en-US/sense/November2018/Subsystems/PlanningQlikSenseDeployments/Content/Sense\\_Deployment/Ports.htm](https://help.qlik.com/en-US/sense/November2018/Subsystems/PlanningQlikSenseDeployments/Content/Sense_Deployment/Ports.htm)

1. Create a mount point on Linux server:

```
# mkdir /qvd-share
```

2. Using the text editor of your choice, create a credentials file containing the username & password of the Windows service account created above:

Example:

```
# vi /root/.credentials
```

Text file should contain the following 2 lines:

```
username=Windows-service-account
```

```
password=service-account-password
```

3. Using the text editor of your choice, EDIT /etc/fstab in order to auto-mount the CIFS share during boot. Append /etc/fstab with the following:

```
//sense1-server-hostname/qvd-share /qvd-share-  
name cifs file_mode=0444,dir_mode=0444,user,credentials=/root/.credentials,rw,uid=500,gid=500,noperm,0 0
```

4. Issue the following command to mount the Windows share:

```
# mount -a
```

5. Issue the following command to verify the contents of the new share:

```
# ls /qvd-share
```

**Note:** Use of CIFS for mounting Windows network shares on Linux servers is well-documented online; there are various different methods for creating these mount points.

## Qlik Sense Virtual Proxy Setup

The following explains how to set up a Qlik Sense Virtual Proxy as part of Qlik Data Catalyst integration.

**Note:** Qlik Sense RootAdmin access is required to perform these operations. This can only be granted to a user after first QMC (Qlik Management Console) login.

**Note:** To export the public and private keys from a Qlik Sense server, please see [https://help.qlik.com/en-US/sense/June2019/Subsystems/ManagementConsole/Content/Sense\\_QMC/export-certificates.htm](https://help.qlik.com/en-US/sense/June2019/Subsystems/ManagementConsole/Content/Sense_QMC/export-certificates.htm)

Upon export, the public key will be in server.pem and the private key will be in server\_key.pem.

1. Generate a JSON Web Token (JWT).

Generate the JSON Web Token (JWT) here: <https://JWT.IO>

- a. Algorithm: [RS256](#)
- b. Payload (remove “sub” and “admin”):

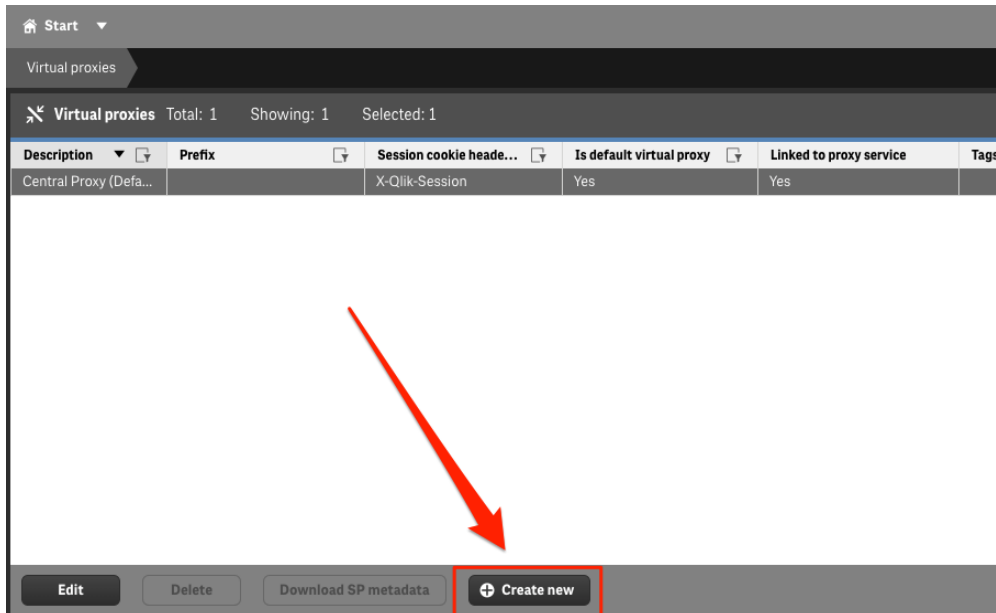
“name”: “[qlik-data-catalyst](#)”

“iat”: [leave as is](#)

- c. Copy text of PUBLIC KEY & paste in first box
- d. Copy text of PRIVATE KEY & paste in second box
- e. JSON Web Token (JWT) is generated in left “Encoded” box
- f. Save the JSON Web Token (JWT) in a text file named “**qdc.jwt**” (*ensure there is no trailing end-of-line after the token in the file*)



### 3. Create a new **Virtual proxy**



Use the following values:

#### Identification Properties

Description: [Qlik Data Catalyst](#)

Prefix: [qdc](#)

Session inactivity timeout (minutes): [30](#)

Session cookie header name: [X-Qlik-QDC-Session](#)

#### Authentication Properties

Anonymous access mode: [No anonymous user](#)

Authentication method: [JWT](#)

JWT Certificate: [<paste the Public certificate that was used to generate the JWT here, server.pem, not the JWT itself>](#)

JWT attribute for user ID: [name](#)

JWT attribute for user directory: [\[QLIK-EXTERNAL-SERVICE\]](#)

**Note:** Public and Private certificates are issued as part of an SSL certificate and are issued for each domain and applied to the Qlik Data Catalyst server. The virtual proxy being created uses JSON Web Token (JWT) based on the SSL certificate; the Qlik Data Catalyst server recognizes it and allows the connection.

4. The proxy configuration screen should look like this:

Click "Apply" to create the virtual proxy.

**IDENTIFICATION**

Description

Prefix

Session inactivity timeout (minutes)

Session cookie header name

**AUTHENTICATION**

Anonymous access mode

Authentication method

JWT certificate

JWT attribute for user ID

JWT attribute for user directory

**JWT attribute mapping**

Static attributes must be enclosed by brackets.

**Properties**

- ✓ Identification
- ✓ Authentication
- Load balancing
- Advanced
- Integration
- Client authentication link
- Tags
- Custom properties

**Proxy restart required**

Any proxies associated with this virtual proxy will be restarted. The sessions for these proxies will be ended and the users logged out. Continue?

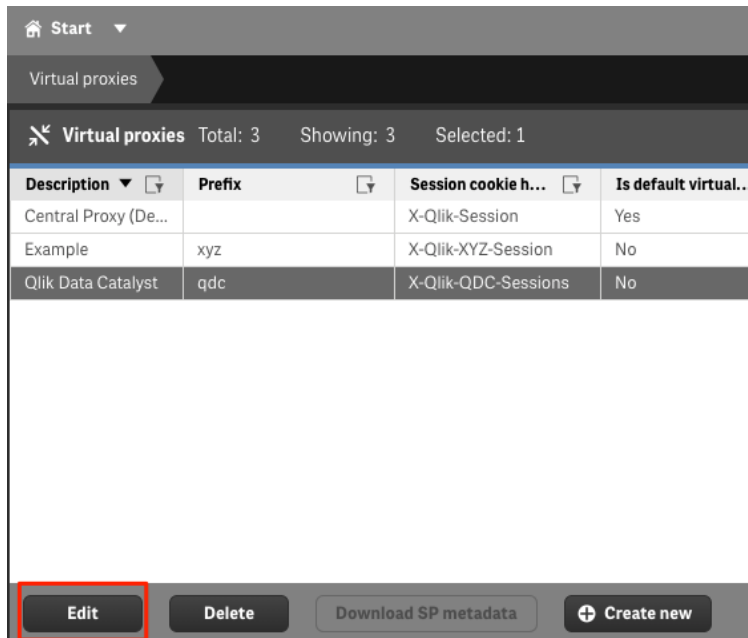
**Note:** Adding the virtual proxy will require a restart of the proxy service which will disconnect any existing Qlik Sense sessions (users will be logged out).

5. Click "OK"



## 6. Link Virtual Proxy to "Central" Proxy:

- Log back into Qlik Management Console (QMC)
- Go back to Virtual Proxies and EDIT the Qlik Data Catalyst virtual proxy



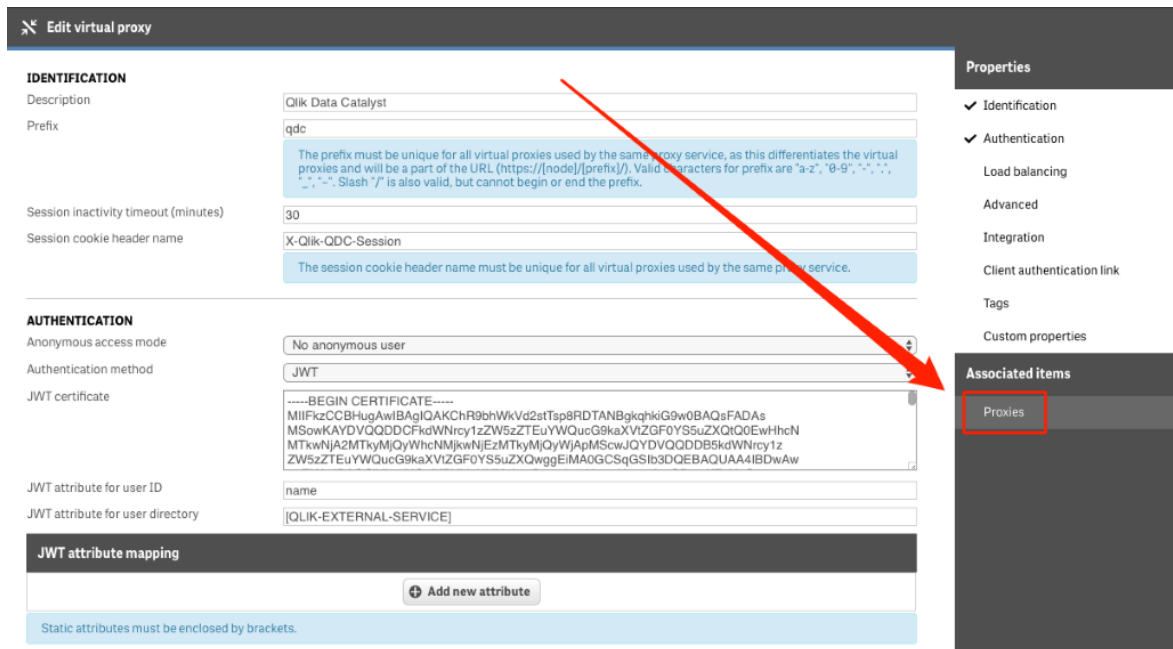
Virtual proxies

Virtual proxies Total: 3 Showing: 3 Selected: 1

Description	Prefix	Session cookie h...	Is default virtual...
Central Proxy (De...		X-Qlik-Session	Yes
Example	xyz	X-Qlik-XYZ-Session	No
Qlik Data Catalyst	qdc	X-Qlik-QDC-Sessions	No

Buttons: Edit, Delete, Download SP metadata, Create new

## c. Associated Items > Proxies



Edit virtual proxy

**IDENTIFICATION**

Description: Qlik Data Catalyst

Prefix: qdc

Session inactivity timeout (minutes): 30

Session cookie header name: X-Qlik-QDC-Session

**AUTHENTICATION**

Anonymous access mode: No anonymous user

Authentication method: JWT

JWT certificate: -----BEGIN CERTIFICATE-----  
MIIFkzCCBHugAwIBAgIQAKChR9bhWkVd2stTsp8RDTANBgkqhkiG9w0BAQsFADAs  
MSowKAYDVQQDDCFkdWNrcy1zZW55ZTEuYWQucG9kaXVlZGF0YS5uZXQ0EwHhcN  
MTkwNjA2MTRyMjQyWWhcNMjkwNjEzMTkyMjQyWjApMScwJQYDVQQDDDB55kdWNrcy1z  
ZW55ZTEuYWQucG9kaXVlZGF0YS5uZXQwggEIMA0GCSqGSIb3DQEBAQUAA4IBDwAw

JWT attribute for user ID: name

JWT attribute for user directory: [QLIK-EXTERNAL-SERVICE]

**JWT attribute mapping**

Static attributes must be enclosed by brackets.

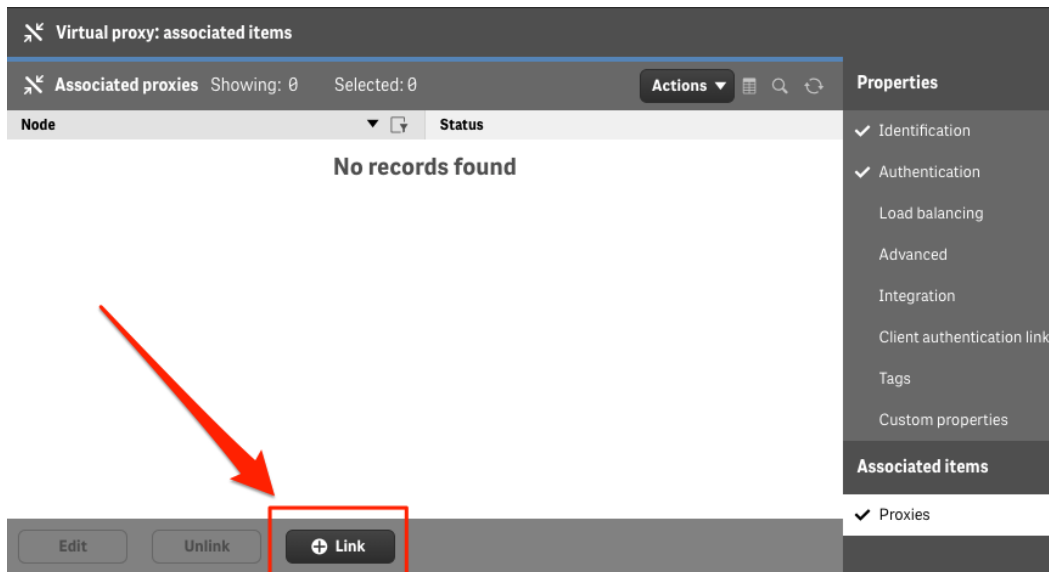
**Properties**

- Identification
- Authentication
- Load balancing
- Advanced
- Integration
- Client authentication link
- Tags
- Custom properties

**Associated items**

- Proxies

d. Click “Link”



e. Select “Central” proxy and then click “Link”



7. On the Qlik Sense server, Admin must run a script to create a user.

**Note:** Local Windows administration permissions are required to perform these steps.

**Note:** *Node.js* must be installed on the Qlik Sense server in order to run the script

- Required files:
  - check-proxy.js
  - qdc.jwt
- Unpack **qdc\_proxy\_artifacts.zip** to extract the **check-proxy.js** file.
- The **qdc.jwt** file created previously contains the JSON Web Token (JWT).
- Edit check-proxy.js script and modify the "**host**" parameter to match the Qlik Sense server URL

```

check-proxy.js - Notepad
File Edit Format View Help
const https = require('https');
const fs = require('fs');

const options = {
  host: 'ducks-sense1.ad.podiumdata.net',
  port: 443,
  method: 'GET',
  rejectUnauthorized: false,
  headers: {}
}

const jwt = fs.readFileSync("qdc.jwt")

const XrfKey = 'yZLGI94HD9zyEZJ8';
options.headers['X-Qlik-XrfKey'] = XrfKey;
options.headers['Authorization'] = 'Bearer ' + jwt;
options.path = '/qdc/qrs/user?xrfkey=' + XrfKey;

```

- Copy the check-proxy.js and qdc.jwt files to the same directory.
- Open a DOS prompt, navigate to the directory with the check-proxy.js script and run it:
 

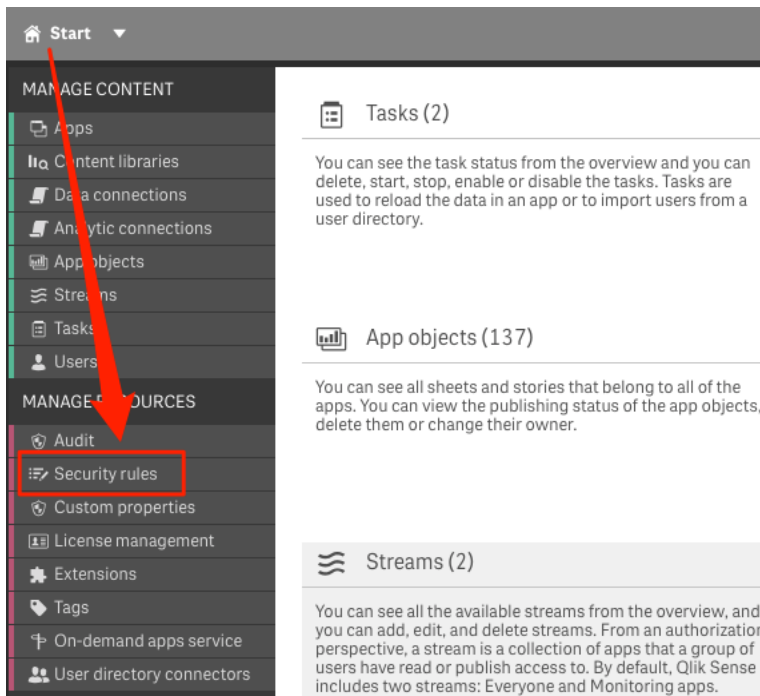
```
node check-proxy.js
```
- If the script runs successfully, you will see a response containing empty brackets “[]”

8. Navigate to QMC (Qlik Management Console) and find the user: **qlik-data-catalyst**

Click Edit > Add role > select “AuditAdmin”

Name	User directory	User ID	Admin roles	Inactive	Blocked	Removed e...
anne	AD	anne		No	No	No
anu	AD	anu	RootAdmin	No	No	No
atif	AD	atif	RootAdmin	No	No	No
carlton	AD	carlton		No	No	No
djenkins	AD	djenkins	RootAdmin	No	No	No
kamemon	AD	kamemon	RootAdmin	No	No	No
kgoraya	AD	kgoraya		No	No	No
kulwinder	AD	kulwinder	RootAdmin	No	No	No
nouser1	AD	nouser1		No	No	No
podium	AD	podium	RootAdmin	No	No	No
qauser1	AD	qauser1		No	No	No
qauser2	AD	qauser2		No	No	No
<b>qlik-data-catalyst</b>	QLIK-EXTERNAL-SERVICE	qlik-data-catalyst	<b>AuditAdmin</b>	No	No	No
raj	AD	raj	RootAdmin	No	No	No
sa_api	INTERNAL	sa_api		No	No	No
sa_converter	INTERNAL	sa_converter		No	No	No
sa_engine	INTERNAL	sa_engine		No	No	No
sa_hub	INTERNAL	sa_hub		No	No	No

9. Navigate to Qlik Management Console => Security Rule =>Create new



When "Edit Security Panel" pops up, enter the following settings:

IDENTIFICATION:

Name: [Security rule for access by QLIK-EXTERNAL-SERVICE](#)

Description: [QLIK-EXTERNAL-SERVICE should have read access to Apps and DataConnections](#)

BASIC:

Resource filter: [DataConnection\\_\\*,App\\_\\*](#)

Actions: [Read](#)

ADVANCED:

Conditions: [\(\(user.userDirectory="QLIK-EXTERNAL-SERVICE"\)\)](#)

Context: [Both in hub and QMC](#)

**IDENTIFICATION**

Disabled

Name Security rule for access by QLIK-EXTER

Description QLIK-EXTERNAL-SERVICE's should have read access to Apps and

**BASIC**

Resource filter DataConnection\_\*,App\_\*

Actions  Create  Read  Update  
 Delete  Export  Publish  
 Change owner  Export data  
 Access offline  Duplicate

user userDirectory = value QLIK-EXTERNAL-SERVICE

**ADVANCED**

Conditions 

```
((user.userDirectory="QLIK-EXTERNAL-SERVICE"))
```

Context Both in hub and QMC

Validate rule

[Link to Qlik Sense help about security rules](#)

**TAGS**

## QVD Import Workflow

To begin importing QVDs, Admins access a **data source** in Qlik Sense. [If needed, refer to Qlik Sense documentation to create a connection and add data: [https://help.qlik.com/en-US/connectors/Subsystems/Integrated\\_Web\\_Connectors\\_help/Content/Connectors\\_QWC\\_BuiltIn/Introduction/Creating-a-connection.htm](https://help.qlik.com/en-US/connectors/Subsystems/Integrated_Web_Connectors_help/Content/Connectors_QWC_BuiltIn/Introduction/Creating-a-connection.htm)]

The following steps detail creation of a Connector in Qlik Data Catalyst and import of QVDs.

1. Login to Qlik Data Catalyst with valid credentials.  
--User must have Admin privileges to access and manage Admin tab.
2. Click on **Admin** on top right-hand side of top task bar
3. Click on **QVD Import** tab
4. Select **Add New Connector**

The screenshot shows the Qlik Data Catalyst interface. At the top, there's a navigation bar with 'Qlik Data Catalyst', 'My Cart', 'Global Search', 'Admin', 'Support', and 'podium'. Below this is a 'QVD Import' section with a 'Refresh Core Env' button. A sidebar on the left contains navigation icons for Ops, Source, Discover, Prepare, Publish, Security, and Catalog. The main content area shows a list of connectors: 'ducks-sense2', 'JRTTestJungleSense', and 'duck\_sense1'. The 'ducks-sense2' connector is selected, and its configuration form is displayed. The form has a title 'QLIK SENSE CONNECTOR' and buttons for 'EDIT', 'DELETE', 'SCHEDULE', and 'VIEW LOGS'. The fields are: Connector Name (ducks-sense2), Default QVD Mount Point (/qvd-write), Host (ducks-sense2.ad.podiumdata.net), Port (0), Username (empty), Proxy (qdc), QDC Base Directory (/usr/local/podium/data), Default Entity Level (MANAGED), Qlik Sense Global Unique ID (875d8d2e-0874-4f23-b8a2-8af2a8273572), and JSON Web Token (JWT) (a long alphanumeric string). At the bottom of the form are 'Cancel', 'Show QVD Paths', 'Test Connector', and 'Save' buttons.

5. On the **QLIK SENSE CONNECTOR** tab enter:

**Connector Name:** Required, User defined

**Default QVD Mount Point:** Required. This value can be entered manually or found in Linux Path when editing the connection in Qlik Sense

**Host:** Required, Qlik Sense Host URL (ex., ducks-sense2.ad.podiumdata.net)

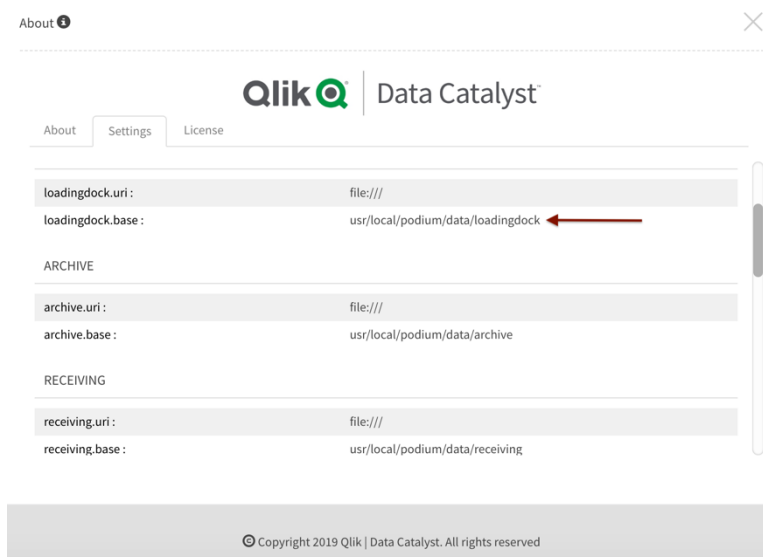
**Port:** Optional (can be skipped)

**Username:** Not currently in use, can be skipped

**Proxy:** Required, Proxy is created in Qlik Sense → QMC → Proxy Section (see [Proxy Setup](#)) Note: Proxy Field is case-sensitive and connection will fail if case does not match. **This is the Identification Prefix entered when setting up the Virtual Proxy – “qdc” was used above.**

**QDC Base Directory:** This is where Qlik Data Catalyst stores the data on local file system. (Copy the base directory information from Support->About->Settings. Copy the value from the "loadingdock.base" property.

(e.g., '/usr/local/podium/data')



**Default Entity Level:** This is populated from System Settings and can be overwritten (options are MANAGED or REGISTERED)

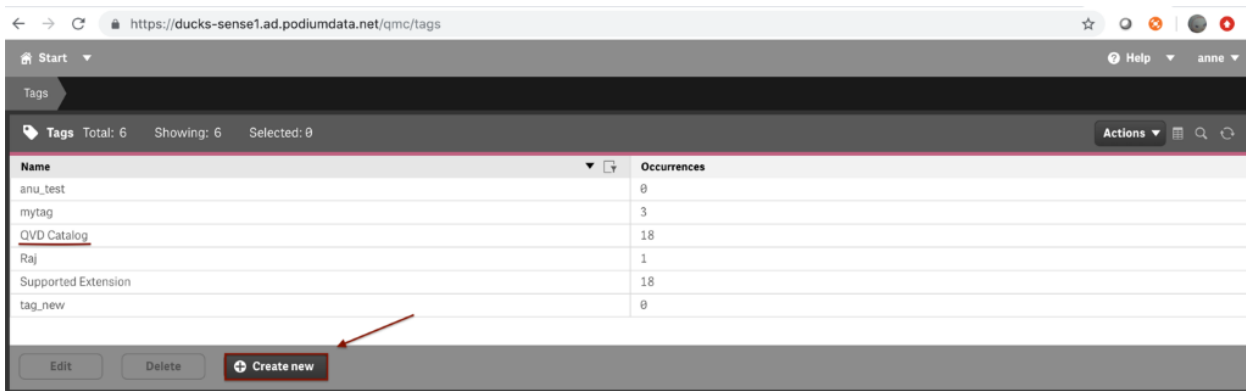
**Qlik Sense Global Unique ID:** Auto-retrieved upon "Test Connection", every installation of Qlik Sense has a Globally Unique Identifier (GUID).

**JSON Web Token (JWT):** This cut-and-paste token is generated as part of Proxy set up [here](#).

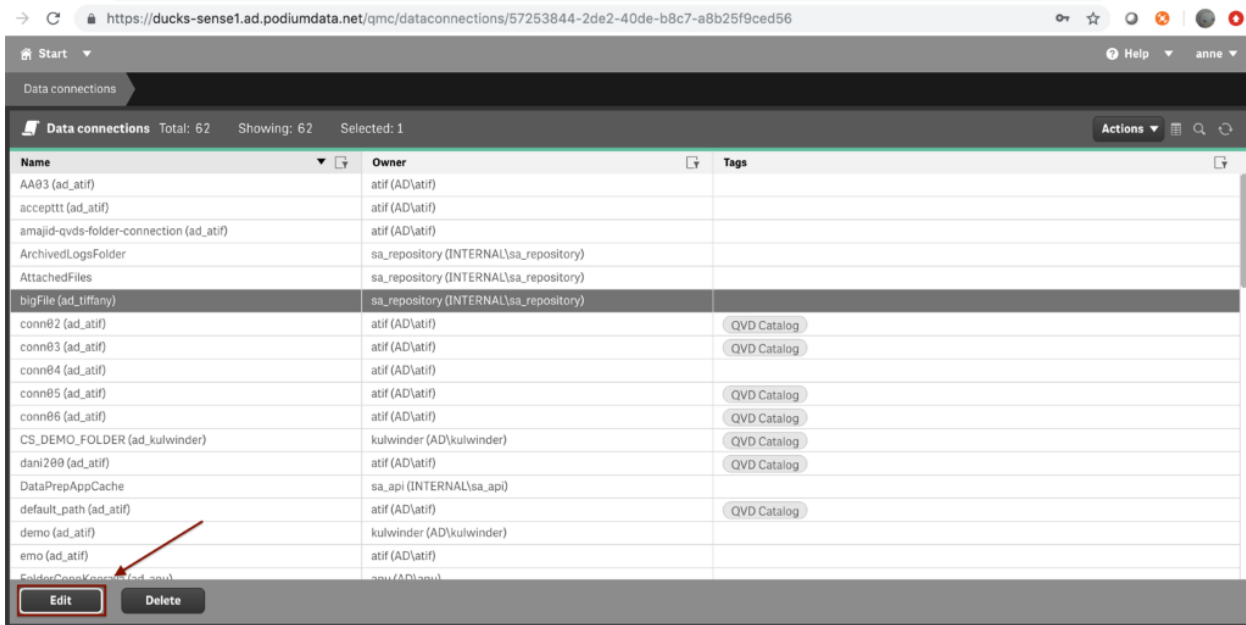
6. Click **Test Connector**. Upon Connection Success, **Save** the Connection
7. Click on **Show QVD Paths** to set up paths
8. In the **QVD Paths** screen, click on **Sync Paths**. All available Qlik Sense connections that have had the "QVD Catalog" tag applied are filtered and imported into Qlik Data Catalyst. When the paths are synced, the Qlik Sense Windows folder must be mapped to the Linux path folder, thereby making Qlik Data Catalyst aware of each QVD in these folders. Every QVD in Qlik Sense corresponds to a new QVD entity in Qlik Data Catalyst.

### Adding "QVD Catalog" Tags in Qlik Sense: QMC (Qlik Management Console)

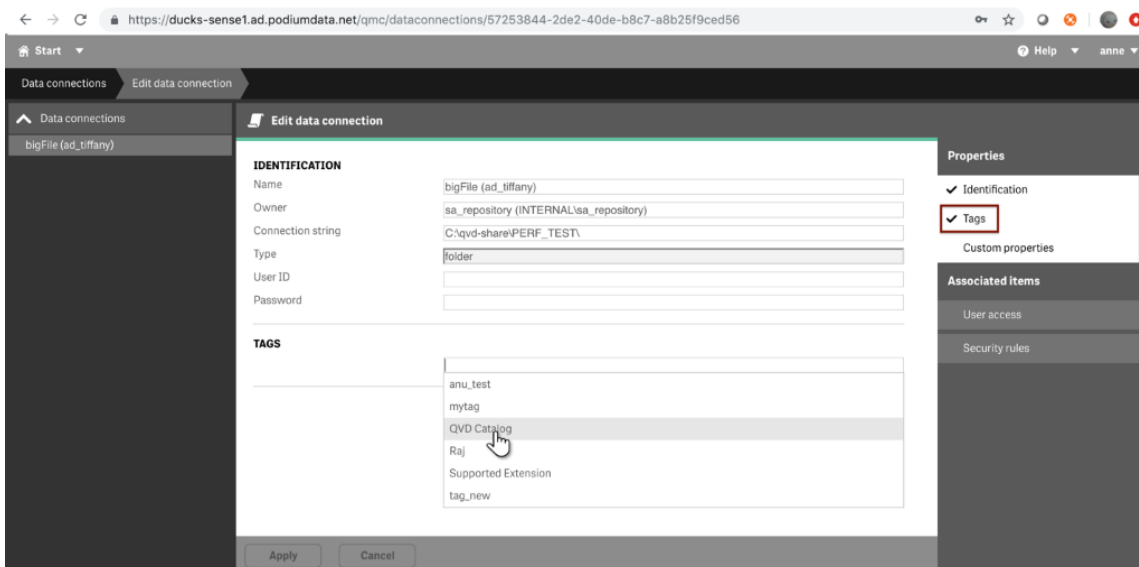
1. Create a "QVD Catalog" Tag in <QlikSenseURL>/qmc/tags



2. Edit a connection in <QlikSenseURL>/qmc/dataconnections



3. Apply the tag to the Connections containing QVDs (that will be imported into Qlik Data Catalyst) in Tags tab

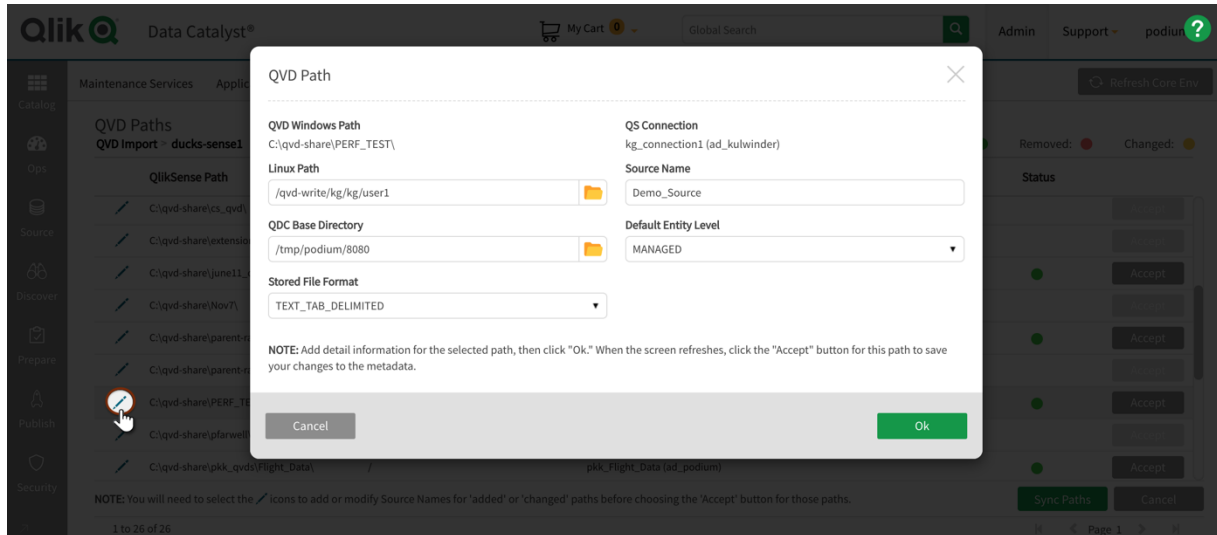




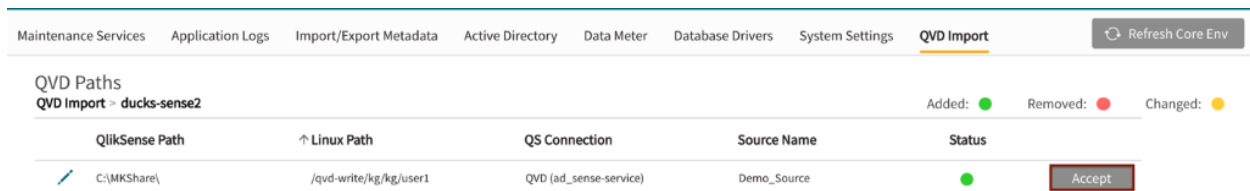
- Click on the pencil icon on any Qlik Sense Path.

Select the Linux Path using the File Browser (this is the mounted Linux path that maps to the corresponding Windows path), then name the Source. QDC Base Directory and Default Entity Level will auto populate from the Connector values but can be overwritten. Click Ok.

**\*Note:** Be sure that the Linux path specified is the mount point for the Windows shared folder containing QVD files on the Qlik Sense server. (Configured previously in [Mount the Qlik Sense Windows Share on the QDC Linux Server: Part 2](#))



- Accept the path to save the metadata. An Admin must select **Accept** to the right of the paths in order to persist the path in Qlik Data Catalyst so that the application knows to scan the folder path and extract information for that folder.



- Go back to QVD Import page. Open the Connector and select **Schedule** and **Run**. The mounted folder (Linux path) is scanned and the QVD entities are added to the created source ("Demo\_Source" in pictured example) created on the Paths screen.

When Run is initiated, Qlik Data Catalyst scans the folder, finds QVDs, and creates/updates/deletes QVDs in Qlik Data Catalyst. File attributes are read from the XML Header of the originating QVD, and information about the QVD required to build a metadata environment (e.g., Fields/Columns) for a QVD entity in Qlik Data Catalyst is extracted in this step.

- Note that data has to be loaded after the metadata environment is established. Users load data for QVD entities like any other Source Type from the Entity grid in the Source Module.

For enhanced security, see the Appendix section "**Configuring Qlik Data Catalyst to Validate a Certificate Issued by an Internal Certificate Authority**".

## Appendix

### Configuring Qlik Data Catalyst to Validate a Certificate Issued by an Internal Certificate Authority

Many Qlik Sense servers are assigned certificates issued by an internal Certificate Authority (e.g., Active Directory), or use self-signed certificates. By default, Qlik Data Catalyst will trust such certificates as the Qlik Sense server is presumed to be within the corporate firewall. To not trust these certificates, and require QDC to validate them, uncomment the property **qlik.trust.all.certs** within **core\_env.properties** and set it to **false**.

When this property is set to false, the SSL certificate presented by the Qlik Sense server will be validated. In order to validate it, you must acquire the CA Root certificate (or self-signed certificate) and add it to the Java runtime's cacerts file. Instructions for this process follow.

### Importing a Certificate Authority (CA) root certificate into the JVM Trust Store

1. *Obtain the root certificate and copy it to the QDC server*

The Qlik Sense “self-signed” root certificate can be found on the Qlik Sense server in the following directory:

```
C:\ProgramData\Qlik\Sense\Repository\Exported Certificates\Local Certificates\root.pem
```

If the Qlik Sense server is using a certificate issue by an internal Certificate Authority, the root certificate must be obtained from the internal Certificate Authority.

2. *Convert the root certificate to DER format*

This can be done with help of the **openssl** toolkit, where root.pem is the original certificate filename in PEM format, and root.der the filename to output, in DER format (which the Java keytool utility can understand).

```
openssl x509 -in root.pem -inform pem -out root.der -outform der
```

3. *Validate the root certificate content*

Ensure that the Java keytool can parse the certificate and display its content:

```
keytool -v -printcert -file root.der
```

4. *Import the root certificate into the JVM trust store*

Enter the following command where \$JAVA\_HOME is a shell environment variable that points to your Java installation:

- The QDC JAVA\_HOME is defined in the Tomcat **setenv.sh** configuration file located in the /bin directory of the Tomcat instance being used by QDC:

```
e.g. /usr/local/qdc/apache-tomcat-7.0.94/bin/setenv.sh
```

- For “alias” pick some unique name for the certificate in the store. e.g. “qliksense” or “internalCA”

```
keytool -importcert -alias qliksense -keystore $JAVA_HOME/jre/lib/security/cacerts -storepass changeit -file root.der
```

(the default password for the CA store is: changeit)

The keytool will prompt you for confirmation, enter yes to complete the operation.

5. *Verify that the root certificate has been imported*

List the trust store content and filter for the certificate alias (name) with grep:

```
keytool -keystore "$JAVA_HOME/jre/lib/security/cacerts" -storepass changeit -list | grep qliksense
```

6. *Restart the QDC Tomcat instance*