

## Admin API

Date published: 2020-10-30

Date modified: 2024-10-30



# Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

# Contents

<b>Admin API.....</b>	<b>4</b>
<b>Admin API syntax and general usage.....</b>	<b>4</b>
<b>Admin API syntax parameters.....</b>	<b>5</b>
<b>Python Admin API data format and response.....</b>	<b>6</b>
<b>CURL data format and API key examples.....</b>	<b>7</b>
<b>Admin API Demo.....</b>	<b>8</b>
Admin API demo examples.....	8
<b>Data type details.....</b>	<b>12</b>
<b>Users.....</b>	<b>13</b>
<b>Groups.....</b>	<b>14</b>
<b>Roles.....</b>	<b>14</b>
<b>Segments.....</b>	<b>17</b>
<b>Filter associations API.....</b>	<b>17</b>
<b>Workspaces.....</b>	<b>18</b>

# Admin API

Cloudera Data Visualization provides a URL-based API, the Admin API, which enables access to Cloudera Data Visualization objects like users, groups, roles, datasets, segments, filter associations, connections, and visuals. This allows you to automate deployment operations, such as creating and managing roles, without logging into the Cloudera Data Visualization server console for access to command-line utilities.

## Permissions

When using the Admin API, Cloudera Data Visualization enforces the same security policies as when using the graphical user interface. For example, you must have Manage roles and users system-level permission to create new users or update existing users.

## Admin API is CRUD

The Admin API supports Create, Read, Update, and Delete (CRUD) operations for basic object access. It does not support complex actions, such as adding tables to datasets.

## Related Information

[Admin API syntax and general usage](#)

[Admin API syntax parameters](#)

[Python Admin API data format and response](#)

[CURL data format and API key examples](#)

[Admin API Demo](#)

[Admin API demo examples](#)

[Data type details](#)

[Users](#)

[Groups](#)

[Roles](#)

[Segments](#)

[Filter associations API](#)

[Workspaces](#)

# Admin API syntax and general usage

The Cloudera Data Visualization Admin API has a consistent pattern for each data type.

The Admin API has the following basic access syntax:

```
[http | https]://host:port/arc/adminapi/version/data_type[/object_id][?options]
```

The parameters of this line are in Admin API syntax parameters.

## HTTP operations

The HTTP method specifies the following operation types:

### GET

List of an item identified through the object\_id or object\_name, or all items, with default summary information. The URL option 'detail=true' returns all data details.

**POST**

Update: The fields in the request data update the item with matching object \_id.

Create: If the URL or request data does not specify the object\_id, ArcViz creates a new data item.

Validate: To verify that the connection is successful, issue the POST command two times:

1. Issue the POST command with the validate flag set to true.

```
[
  {
    "id": 18,
    "name": "ArcEngine Dev",
    "type": "arcengine",
    "validate": "true",
    "info": {
      "PARAMS": {
        "HOST": "localhost",
        "PORT": "21051",
        "USERNAME": "admin",
        "SETTINGS": {
          "ANALYTICAL_VIEW_MAX_REFRESH_THREADS": "1",
          "MAX_PARTITIONS_FOR_REFRESH_INSERT": "1"
        }
      }
    }
  },
  .....
  .....
]
```

2. On success, issue the same POST command without the validate flag. This step saves the data.

**DELETE**

Delete the specified item.

**HTTP access through Python**

While you can use all standard HTTP access methods, we recommend the python request modules approach for HTTP operations. Note the following common setup:

```
import json
import requests
api_url = [http|https]://host:port/arc/adminapi/version
login_url = [http|https]://host:port/arc/apps/login
```

**Related Information**

[Admin API syntax parameters](#)

## Admin API syntax parameters

The Cloudera Data Visualization Admin API has a consistent pattern for each data type.

The Admin API has the following basic access syntax:

```
[http | https]://host:port/arc/adminapi/version/data_type[/object_id][?options]
```

**host**

The host of the Cloudera Data Visualization instance.

**port**

The port of the Cloudera Data Visualization instance.

**version**

The current API version is v1. This increments if the item data format changes. Whenever possible, we intend to support older versions for backward compatibility.

**data\_type**

One of the Cloudera Data Visualization artifacts: users, groups, roles, datasets, connections, visuals, segments, filter associations, or workspaces.

**object\_id**

The id of the individual object, such as a specific user, visual, or a specific dataset. You can either use the object\_id or the object\_name in the syntax, not both.

**object\_name**

The name of the individual object, such as a specific user, visual, or a specific dataset. You can either use the object\_id or the object\_name in the syntax, not both.

**options**

Further request options, such as level of information details when 'detail=true'.

**api\_url**

The address of the API management system, in the form [http|https]://host:port/arc/adminapi/version.

**login\_url**

To authenticate the end user, address of the login URL, in the form [http|https]://host:port/arc/apps/login.

## Python Admin API data format and response

Cloudera Data Visualization provides examples of python Admin API data format and response.

The response data for GET operations is a list of JSON items. For POST operations, such as UPDATE and CREATE, the input format is a structure with a data field that contains the JSON list of items. The UPDATE and CREATE operations process one item at a time, so the list is exactly one entry long.

The response data for POST operations is the updated item, matching a GET with detail=1 for the item, as demonstrated in [Example 1](#).

For item updates, it is only necessary to specify the fields you are updating. Cloudera Data Visualization merges the supplied fields in the input data with the existing item's data, as demonstrated in [Example 2](#).

### Example 1: Setting the name for role ID=1 to 'System Admin'

```
payload = {'name': 'System Admin'}
session.post(api_url + '/roles/1', data={'data': json.dumps([payload])})
```

Note that the API URL has the following form:

```
[http|https]://host:port/arc/adminapi/version
```

For syntax of other parameters, see Admin API syntax parameters.

### Example 2: Checking role ID=2; updating by adding a new user

```
response = session.get(api_url + '/roles/2?detail=1')
```

```
role = response.json()[0]
if 'new_user' not in role['users']:
    payload = {'users':role['users'] + ['new_user']}
    session.post(api_url + '/roles/2', data={'data':json.dumps([payload])})
```

For the definition of fields for each data type, see [Data type details](#).

### Related Information

[Admin API syntax parameters](#)

[Data type details](#)

## CURL data format and API key examples

Cloudera Data Visualization provides examples of API Key in CURL data format.

When you add the APIKey to the request header and avoid explicit login, all interactions become simpler. The examples in this article use an APIKey obtained through the Manage API Keys interface, on the host:port/arc/apps/apikeys browser page of the DataViz installation. The actual APIKey and the method of retrieving the key depends on the user system.

See [Example 1](#) to learn how to get all roles information and [Example 2](#) to change the description of a role.

### Example 1: Getting all roles

To use CURL to obtain a full dump of all roles, use the following command.

Note that the output is piped to the standard python JSON dumper for easier reading; it is not necessary for CURL access.

```
curl -s \
-X GET \
-H "Authorization: apikey ApiKey" \
api_url/roles?detail=1> | python -m json.tool
```

Note that the login URL has the form [http|https]://host:port/arc/apps/login. For syntax of other parameters, see [Admin API syntax parameters](#).

### Example 2: Changing the role description

To change the description for role ID 3, use the following command.

For CURL to supply data through the POST method, use the application/x-www-form-urlencoded content type.

```
curl -s \
-X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "Authorization: apikey ApiKey" \
-d 'data=[{"desc":"Updated description again"}]' \
api_url/roles/3
```

For the definition of fields for each data type, see [Data type details](#). For syntax of other parameters, see [Admin API syntax parameters](#).

### Related Information

[Admin API syntax parameters](#)

[Data type details](#)

## Admin API Demo

This is a simple GUI demo to demonstrate how the Cloudera Data Visualization URL-based Admin API can easily display the data format for each item type.

**Note:**

- Use the demo with some caution: the system does not confirm DELETE operations, and it is not meant for implementing system configuration changes. It is simply a working example of the Admin API.
- The demo is not enabled by default. It must be enabled, much as the actual Admin API URL support must be enabled, for each item.

Use the following statement to enable the demo:

```
ADMIN_API_DEMO_LIST = ['visuals', 'datasets', 'connections', 'users', 'groups',  
    'roles', 'segments', 'filterassociations']
```

Alternatively, use the wild card to specify all options:

```
ADMIN_API_DEMO_LIST = ['*']
```

To fully enable all APIs and all demo tabs, use the wild card character in the settings entries for both ADMIN\_API\_DEMO\_LIST and ADMIN\_API\_URL\_LIST:

```
ADMIN_API_DEMO_LIST = ['*']  
ADMIN_API_URL_LIST = ['*']
```

After enabling the demo, you can access it through the following URL:

```
host_path/arc/apps/apidemo
```

## Admin API demo examples

Cloudera Data Visualization's URL-based Admin API provides a straightforward way to display the data format for each item type.

The demo has a tab for each enabled item type, and supports all the API functions: Fetch (one), Fetch All, Create, Update, and Delete.

The following examples demonstrate some of the functions available through the demo.

### Fetching all user information

To get information on all users, click Fetch. Cloudera Data Visualization returns the list of registered users, as shown in the following screenshot.



**CLOUDERA**  
Data Visualization

[HOME](#)
[SQL](#)
[VISUALS](#)
[DATA](#)

[Visuals](#)
[Datasets](#)
[Connections](#)
[Users](#)
[Groups](#)
[Roles](#)
[Segments](#)
[Filter Associations](#)
[Workspaces](#)

FETCH

ID/Name

Detail ☐

URL: GET /arc/adminapi/v1/users

```

[
  {
    "id": 1,
    "username": "vizapps_admin",
    "is_superuser": true
  },
  {
    "id": 2,
    "username": "Administrator",
    "is_superuser": false
  },
  {
    "id": 3,
    "username": "DataVizAdmin",
    "is_superuser": false
  },
  {
    "id": 4,
    "username": "DataVizUser",
    "is_superuser": false
  },
  {
    "id": 5,
    "username": "Rep1",
    "is_superuser": false
  },
  {
    "id": 6,
    "username": "Rep2",
    "is_superuser": false
  }
]

```

### Fetching single user information, with detail

To extract information on a single item using its ID or name, follow these steps:

1. Under the Users tab, enter the ID.
2. Select the Detail option to get the full data for that item, in this case user ID 1.
3. Click Fetch.

FETCH

UPDATE

DELETE

ID/Name

Detail ☐

URL: GET /arc/adminapi/v1/users/1

```

[
  {
    "id": 1,
    "username": "vizapps_admin",
    "is_superuser": true
  }
]

```

### Cloning an item

When you clone an item, the resulting screen shows a duplicate of the item but clears the ID. field. For more information, see [Creating a new item](#) on page 10.

**CLOUDERA**  
 Data Visualization

[HOME](#)
[SQL](#)
[VISUALS](#)
[DATA](#)

[Visuals](#)
[Datasets](#)
[Connections](#)
[Users](#)
[Groups](#)
[Roles](#)
[Segments](#)
[Filter Associations](#)

FETCH
 CREATE
 ID 
 Detail URL: POST /arc/adminapi/users

```

{
  {
    "username": "user2",
    "is_superuser": false,
    "is_active": true,
    "date_joined": "2014-12-08 22:27:27 UTC",
    "last_login": "2017-04-06 02:06:21 UTC",
    "groups": [],
    "roles": [
      {
        "id": 5,
        "name": "For user2"
      }
    ]
  }
}
  
```

### Creating a new item

If you click CREATE immediately after cloning an item, Cloudera Data Visualization returns an error because a user with this name already exists.

To create a new user, change the username and provide an initial password. Here is an example in which a new user was created by changing the username and adding an extra line that specifies the password:

```

"username": "user2-copy",
"password": "initial-pw",
  
```

**CLOUDERA**  
 Data Visualization

[HOME](#)
[SQL](#)
[VISUALS](#)
[DATA](#)

[Visuals](#)
[Datasets](#)
[Connections](#)
[Users](#)
[Groups](#)
[Roles](#)
[Segments](#)
[Filter Associations](#)

FETCH
 CREATE
 ID 
 Detail URL: GET /arc/adminapi/users/3?detail=1

```

{
  {
    "username": "user2-copy",
    "password": "initial-pw",
    "is_superuser": false,
    "is_active": true,
    "date_joined": "2014-12-08 22:27:27 UTC",
    "last_login": "2017-04-06 02:06:21 UTC",
    "groups": [],
    "roles": [
      {
        "id": 5,
        "name": "For user2"
      }
    ]
  }
}
  
```

Click Create again. Cloudera Data Visualization notifies you that the update is successful and refreshes the interface to show the results, consistent with fetching detailed information for a specific user, as described in [Fetching single user information, with detail](#) on page 9.



**Note:** For security reasons, the password is not included in the fetched user details.

The screenshot shows the Cloudera Data Visualization Admin API Demo interface. At the top, there's a dark navigation bar with the Cloudera logo and 'Data Visualization' text, and links for HOME, SQL, VISUALS, and DATA. Below this, a light blue header contains 'Visuals', 'Datasets', 'Connections', and 'Filter Associations'. A green success message box in the center reads 'Success Update Success!'. Below the header, there are buttons for 'FETCH', 'UPDATE', 'CLONE', and 'DELETE', followed by a text input field containing 'ID 12'. To the right of these buttons are links for 'Detail' (with a blue icon) and 'URL: POST /arc/adminapi/users'. The main content area displays a JSON object representing user details:

```
{
  {
    "id": 12,
    "username": "user2-copy",
    "is_superuser": false,
    "is_active": true,
    "date_joined": "2017-04-09 00:39:59 UTC",
    "last_login": "2017-04-09 00:39:59 UTC",
    "groups": [],
    "roles": [
      {
        "id": 5,
        "name": "For user2"
      }
    ]
  }
}
```

## Changing passwords

To change a user's password, you must supply both the current password (password) and the new password (new\_password).

Edit the user detail by adding the following lines of code and click UPDATE.

```
"password": "initial-pw",
"new_password": "updated-pw",
```

The screenshot shows the Cloudera Data Visualization interface. At the top is a dark navigation bar with the Cloudera logo and links for HOME, SQL, VISUALS, and DATA. Below this is a secondary navigation bar with tabs for Visuals, Datasets, Connections, Users (selected), Groups, Roles, Segments, and Filter Associations. Under the Users tab, there are buttons for FETCH, UPDATE, CLONE, and DELETE, followed by a text input field containing 'ID 12'. To the right of the input field is a 'Detail' link with a blue icon and the URL 'POST /arc/adminapi/users/12'. The main content area displays a JSON object representing a user:

```
{
  {
    "id": 12,
    "username": "user2-copy",
    "password": "initial-pw",
    "new_password": "updated-pw",
    "is_superuser": false,
    "is_active": true,
    "date_joined": "2017-04-09 00:39:59 UTC",
    "last_login": "2017-04-09 00:39:59 UTC",
    "groups": [
      {
        "id": 6,
        "name": "Clone of group Test-0"
      }
    ],
    "roles": [
      {
        "id": 5,
        "name": "For user2"
      }
    ]
  }
}
```

## Data type details

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server objects.

The Admin API uses specific JSON definitions for each data type:

- Users
- Groups
- Roles
- Segments
- Filter associations API
- Workspaces

Note that we deliberately chose not to document the details of creating datasets, connections, and visuals. They are all highly complex structures, and should be created directly in the Cloudera Data Visualization application, through the graphical user interface.

For the GET requests, many returned fields are only informational, and cannot be updated through subsequent POST requests.

Some fields are themselves complex structures that contain sub-fields. The update logic that merges the supplied input fields with the existing data applies to the top-level fields only. For example, the role type contains a privs field, which is a list of individual privilege records. To update the privs field, you must supply the entire list, not merely the individual list element.

### Related Information

[Users](#)

[Groups](#)

[Roles](#)

[Segments](#)  
[Filter associations API](#)  
[Workspaces](#)

## Users

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server object, users.

When creating a new user, you must supply the password field. You must also supply the password field when updating a user's password. Like in the GUI, a regular user (non-admin) can only change their own password, and must supply the current password in the old\_password field.

Supplying None (or null in the API demo page, the javascript version of None) for the password makes the user account unusable for login.

For the list of groups and roles, the name is supplied for information only. When updating the users's groups or roles, only the ID fields are necessary.

The examples in this article use an API Key obtained through the Manage API Keys interface, on the host:port/arc/apps/apikeys browser page of the DataViz installation. The actual API Key and the method of retrieving the key depends on the user system.

Here is a CURL example for setting the roles for user ID=3 to IDs 5, 7, and 8.

```
curl -s \
-X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "Authorization: apikey api_key" \
-d 'data=[{"roles": [{"id":5}, {"id":7}, {"id":8}] }]' \
api_url + '/users/3'
```

Note that the API URL has the form [http|https]://host:port/arc/adminapi/version.

The JSON fields for user's data type are defined as follows:

**Table 1: JSON Fields for Users Data Type**

Field	Detail Only	Updatable	Description
id	No	No	User ID
username	No	Yes	Username, limited to alphanumeric, period, underscore, and dash
is_superuser	No	No	Indicates the admin userm, who has full permissions
is_active	Yes	No	TRUE: user can long through the UI FALSE: user cannot login through the UI, but trusted authentication works
date_joined	Yes	No	Shows the creation date for this user's metadata entry
last_login	Yes	No	Shows the last login date for this user
groups	Yes	Yes	List of groups to which this user belongs; each entry shows group ID and group name
roles	Yes	Yes	List of roles to which this user belongs; each entry shows role ID and role name

## Groups

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server object, groups.

Like the users data type, the names for list of groups and roles are supplied for information only. When updating the users's groups or roles, only the ID fields are necessary.

The JSON fields for group's data type are defined as follows:

**Table 2: JSON Fields for Groups Data Type**

Field	Detail Only	Updatable	Description
id	No	No	Group ID
name	No	Yes	Group name
users	Yes	Yes	List of users in this group; each entry shows the user id and username
roles	Yes	Yes	List of roles to which this group belongs; each entry shows role ID and role name

## Roles

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server object called roles.

To support installations that store the users and groups information outside the Cloudera Data Visualization environment (such as LDAP), the role membership lists for users and groups only store names. During role update and create operations, Cloudera Data Visualization accepts user and group names 'as is', without validating them.

Each entry in the privs list corresponds to a single privilege row on the Cloudera Data Visualization role edit page. Each row contains fields for the privilege type (ptype), an identifier section, and a list of permissions (perms) for the identified objects, such as datasets or data connections. Each privilege type has a specific identifier, and set of possible permissions. Cloudera Data Visualization stores the dataset IDs and connection IDs within the identifier sections as a STRING and uses the special value -1 to indicate 'All dataset' or 'All connections'.

### Roles data type

The JSON fields for the roles' data type are defined as follows:

**Table 3: JSON fields for the roles data type**

Field	Detail Only	Updatable	Description
id	No	No	Role ID
name	No	Yes	Role name
desc	No	Yes	Role description
users	No	Yes	List of usernames that belong to this role
groups	No	Yes	List of groups that belong to this role
privs	Yes	Yes	List of privilege structures for this role, as described in <a href="#">Privileges types</a> on page 14

### Privileges types

The Cloudera Data Visualization Role-Based Access Control system supports the following permission types:

- *ptype: "system"*

**Identifier**

None

**Permissions**

Permission name	Description
sys_editperm	Manage roles and users
sys_styles	Manage styles and settings
sys_viewlogs	View query logs
sys_editconn	Manage data connections

- *ptype: "dataconn"*

**Identifier**

Field name	Description	Example
dclist	List of data connection IDs, or -1 for 'All data connections'	"dclist" : ["-1"]

**Permissions**

Permission name	Description
dc_aviews	Manage analytical views
dc_upload	Import data
dc_expore	Create datasets and explore tables

- *ptype: "dataset"*

**Identifier**

Field name	Description	Example
dcid	Data connection ID for this privilege, or -1 for 'All'	"dcid" : "-1"
dslist	List of dataset IDs for this privilege	"dslist" : ["1", "2", "3"]

**Permissions**

Permission name	Description
dc_aviews	Manage analytical views
dc_upload	Import data
dc_expore	Create datasets and explore tables

**Creating roles**

The following code creates a new role with groups dataconn\_managers and arcviz\_admins. The role has system-level permissions to view logs, and to create new datasets. It also has full permissions on all connections and all datasets.

The actual API Key and the method of retrieving the key depends on the user system.

```
curl -s \
-X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "Authorization: apikey api_key" \
-d 'data=[{
  "name": "Connection manager",
  "desc": "Data connection management",
  "groups": ["dataconn_managers", "arcviz_admins"],
```

```

    "privs": [
      {
        "ptype": "system",
        "perms": ["sys_viewlogs", "sys_editconn"]
      },
      {
        "ptype": "dataconn",
        "dclist": ["-1"],
        "perms": ["dc_aviews", "dc_upload", "dc_explore"]
      },
      {
        "ptype": "dataset",
        "dcid": "-1",
        "dslist": ["-1"],
        "perms": ["ds_manage", "ds_appedit", "ds_appview"]
      }
    ]
  }
}]]' \
127.0.0.1:7999/arc/adminapi/roles

```

When viewing this role through the Cloudera Data Visualization user interface, it appears on the edit page like this:

Role: Connection manager

Name: Connection manager

Description: Data connection management

Privilege Members

System	Active	Additional system privilege
Create workspaces	Allows users to create new workspaces that may be shared among users and user groups.	Users can set a default homepage, manage dashboards in other users' private workspaces, perform administrative restart/stop operations, and request tickets from the Cloudera Data Visualization Server using Trusted Auth Get Ticket.
View roles and users	Enables users to view users, user groups, and roles.	
Manage roles and users	Enables users to create users, user groups, and roles.	
Manage settings	Permits users to manage global site settings.	
Manage custom styles	Authorizes users to create new styles for dashboards and visuals.	
Manage jobs, email templates	Enables users to manage scheduled jobs and create templates for email messages.	
View activity logs	Allows users to view usage statistics and monitor the performance of Cloudera Data Visualization.	
Manage data connections	Grants users the ability to create and manage connections to various data sources.	

Privilege Members

System: Active

Roles: Active

Connections: Active

Datasets: Active

ADD CONNECTIONS

Manage AVs/Extracts	Import data	Create datasets, explore tables
Enables users to create and manage analytical views and data extracts.	Allows users to import supplemental data into an existing connection.	Allows users to create new datasets from existing tables, view sample data, and explore statistical reports on the data tables.
All connections		

Privilege Members

System: Active

Roles: Active

Connections: Active

Datasets: Active

ADD DATASETS

Manage dataset	Manage dashboards	View dashboards
Allows users to change the properties of datasets, create datasets over joined tables, modify the fields of the dataset, and more.	Enables users to create and modify visuals and dashboards.	Used to limit users to view-only privileges for visuals and dashboards, while denying edit privileges.
All connections / All datasets		



## Segments

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server object, segment.

The segment data field is a complex structure that matches the UI Segment edit screen.

This article includes the following topics:

- [Segment Data Type](#) on page 17
- [Data Field Detail](#) on page 17

### Segment Data Type

The JSON fields for role's data type are defined as follows:

**Table 4: JSON Fields for Segment Data Type**

Field	Detail Only	Updatable	Description
id	No	No	Segment ID
name	No	Yes	Segment name
dataset_id	No	No	Dataset ID for the segment
created	Yes	No	Shows the creation date for this segment
created_by	Yes	No	Username of the segment creator
updated	Yes	No	Shows the most recent update for this segment
updated_by	Yes	No	Username of the segment updater
data	Yes	Yes	Segment definition data, as described in <a href="#">Data Field Detail</a> on page 17

### Data Field Detail

The Cloudera Data Visualization segment data field is a complex structure with the following specification:

Field	Description
entities	List of dimension expressions emitted by the entity-type segments
group	Name of the group to which this segment belongs
filters	List of filter expressions that define this segment
applyToNewVisuals	Specify if new visuals on the dataset should start with filters defined in this segment

## Filter associations API

Cloudera Data Visualization provides URL access to the Cloudera Data Visualization server object, filter association.

The filter association data field is a complex structure that matches the Filter Association edit screen.

This article includes the following topics:

- [Segment Data Types](#) on page 18
- [Data Field Detail](#) on page 18

## Segment Data Types

The JSON fields for role's data type are defined as follows:

**Table 5: JSON Fields for Filter Association Data Type**

Field	Detail Only	Updatable	Description
id	No	No	Filter association ID
name	No	Yes	Filter association name
dataset_id	No	No	Dataset ID for the filter association
created	Yes	No	Shows the creation date for this filter association
created_by	Yes	No	Username of the filter association creator
updated	Yes	No	Shows the most recent update for this filter association
updated_by	Yes	No	Username of the filter association updater
users	Yes	Yes	List of user IDs to which the filter association applies
groups	Yes	Yes	List of group IDs to which this filter association applies
data	Yes	Yes	List of segments that make up this filter association, as described in <a href="#">Data Field Detail</a> on page 18

## Data Field Detail

The Cloudera Data Visualization filter association data field is a complex structure with the following specification:

Field	Description
id	ID of segment applied to filter association
group	Name of the group to which the identified segment belongs
negate	Indicates that the filter association defines the rows NOT IN the segment, rather than IN the segment

# Workspaces

Cloudera Data Visualization provides URL access to the ArcViz server object, workspaces.

In addition to the standard public workspace that all users share, and the single private workspace that each user has, Cloudera Data Visualization users with Create Workspace privilege can create custom workspaces. These workspaces may be shared by specific users and user groups, so they can develop and view dashboards that are inherently useful to their line of business. Within each of these workspaces, each user has a defined access level: View Only, Edit, or Manage.

This article includes the following topics:

- [Workspace data type](#) on page 18
- [Access control list in workspaces](#) on page 19
- [Creating workspaces](#) on page 19

## Workspace data type

The JSON fields for workspace data type are defined as follows:

**Table 6: JSON fields for workspace data type**

Field	Detail Only	Updatable	Description
id	No	No	Workspace ID
name	No	Yes	Workspace name
desc	No	Yes	Workspace description
editable	Yes	No	Permission to update only a non system-managed workspace (Custom workspace). The system managed workspaces are Public and Private workspaces.
private_user_id	Yes	No	ID of the user for private workspaces
acl	Yes	Yes	Access Control List (ACL) for a workspace

### Access control list in workspaces

The workspace ACL is a list of privilege entries. Each entry contains three items. The following acl syntax shows two entries:

```
"acl": [
  [entry_type, access_level, group_name],
  [entry_type, access_level, user_name]]
```

Entry	Encoding
entry_type	1 = User, 2 = Group
access_level	1 = View, 2 = Edit, 3 = Manage
user_name/group_name	User or group name of the entry_type

### Creating workspaces

The following code creates a new workspace Test workspace and provides View access to a special group Everyone and Manage access to user admin.

The actual APIKey and the method of retrieving the key depends on the user system.

```
curl -s \
-X POST \
-H "Content-Type: application/x-www-form-urlencoded" \
-H "Authorization: apikey apikey" \
-d 'data=[{
  "name": "Test workspace",
  "desc": "Workspace created via admin api",
  "acl": [[2, 1, "Everyone"], [1, 3, "admin"]]
}]' \
127.0.0.1:7999/arc/adminapi/workspaces
```

When viewing this workspace through the Cloudera Data Visualization UI, it appears on the workspace edit modal window like this:


Create Workspace ✕

Name


Test Workspace

Description

Workspace created via admin api




type username



type group name, type 'Everyone' for all

ADD




admin

☐ View Only

☐ Edit

☒ Manage

✕



Everyone


☒ View Only

☐ Edit

☐ Manage

✕

NOTE: Users must also have appropriate privileges to the underlying dataset(s)

 DELETE WORKSPACE

CLOSE

SAVE