Cloudera Octopai Data Lineage 1.0.0

API

Date published: 2025-10-09
Date modified: 2025-10-20

CLOUDERA

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. 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

Cloudera Data Lineage API 4
Authentication 4
Cloudera Data Lineage Extraction APIs 5
Cloudera Data Lineage API: UserEvents documentation for user audit trails 11

Cloudera Data Lineage API

The Cloudera Data Lineage (formerly Octopai) API enables developers to integrate Cloudera Data Lineage seamlessly into their workflows, enabling efficient and secure data operations.

Authentication

The Cloudera Data Lineage (formerly Octopai) API ensures secure communication through HTTPS protocols and JSON Web Token (JWT) Bearer authentication with refresh tokens.

The Cloudera Data Lineage API supports the following security protocols:

- JWT-based Bearer authentication
- HTTPS enforced for all endpoints to ensure secure communication

Use the following API request to get access_token:

- **Endpoint**

```
POST https://[***COMPANY-NAME***].octopai.com/api/UserAccount/Login
```

- **Request body**

```
{"Username": "[***USER-EMAIL***]","Password": "[***PASSWORD***]"}
```

The request body has the following parameters:

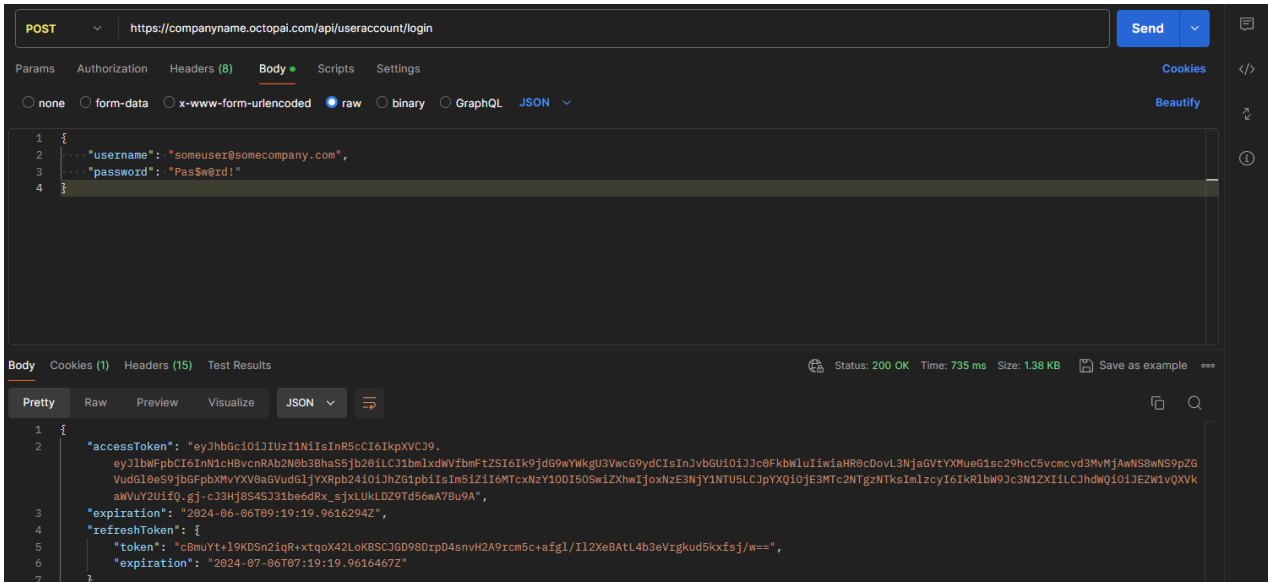
Parameter	Type	Description	Example
Username	string	Octopai user name	john.doe@example.com
Password	string	Octopai user password	Pas$w@rd

- **Sample response**

```
{
"accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFpbCI6InN1cHBvc
nRAb2N0b3BhaS5jb20iLCJ1bmlxdWVfbmFtZSI6Ik9jdG9wYWkgU3VwcG9ydCIsInJvbGUiO
iJJc0FkbWluIiwiaHR0cDovL3NjaGVtYXMueG1sc29hcC5vcmcvd3MvMjAwNS8wNS9pZGVud
Gl0eS9jbGFpbXMvYXV0aGVudGljYXRpb24iOiJhZG1pbiIsIm5iZiI6MTcxNzY1ODI5OSwiZ
XhwIjoxNzE3NjY1NTU5LCJpYXQiOjE3MTc2NTgzNTksImlzcyI6IkRlbW9Jc3N1ZXIiLCJhd
WQiOiJEZW1vQXVkaWVuY2UifQ.gj-cJ3Hj8S4SJ31be6dRx_sjxLUkLDZ9Td56wA7Bu9A",
"expiration": "2024-06-06 09:19:19",
"refreshToken": {
        "token": "cBmuYt+l9KDSn2iqR+xtqoX42LoKBSCJGD98DrpD4snvH2A9rcm5c+a
fgl/Il2XeBAtL4b3eVrgkud5kxfsj/w==",
        "expiration": "2024-07-06 07:19:19"
    },
    "userName": "john.doe@example.com",
    "displayName": "John Doe",
    "userEmail": null,
    "isAdmin": true,
    "error": null,
    "groupMember": [],
    "authLevel": "ADMIN",
    "displayModules": "lineage;e2e;discovery;compare;adc;adc dashboard;liv
elineage"
}
```

Use access_token for all following API requests.

Postman example



Cloudera Data Lineage Extraction APIs

Learn about the API endpoints that are designed to facilitate the extraction of lineage data from the Cloudera Data Lineage (formerly Octopai) system.

For optimal results, start with a call to /assets/query to obtain a comprehensive list of assets. Then call /lineage to retrieve detailed lineage information for each asset identified in the initial query. This two-step approach delivers a structured and efficient data retrieval flow and provides clear insight into asset lineage within the Cloudera Data Lineage ecosystem.

Assets API

This API retrieves data about assets such as columns and tables so you can review their properties.

Use this API to locate the assets _key value that is required when you call /lineage.

Key function

Search assets.

Endpoint

POST /api/v2.0/assets/query

Request body

```
{
  "ConnectionIds": ["101", "102"],
  "AssetNames": ["storeid", "customer"],
  "ToolNames": ["SQLS", "POWERBI"],
  "ToolTypes": ["DB", "ETL"],
  "AssetPath": "AdventureWorks2014.dbo",
  "AssetType": 1,
  "Limit": 1000,
  "IsFaked": false,
  "Offset": 0
}
```

Request body parameters

Parameter	Required	Type	Description
ConnectionIds	No	list	Search by connection identifiers. Example: ["101","102","103"]
AssetNames	No	list	Match assets that contain one of the supplied names.
ToolNames	No	list	Filter by tool names.
ToolTypes	No	list	Filter by tool type. Set assetType to 1 for cross-system lineage (DATABASE / ETL / REPORT). Set assetType to 2 for column-level lineage (DB / ETL / REPORT).
DatabaseName	No	string	Filter by database name.
SchemaName	No	string	Filter by schema name.
LayerName	No	string	Filter by layer name.
TableName	No	string	Filter by table name. **Note:** The TableName parameter is only applicable to column-level lineage.
assetType	Yes	number	1 – Cross-system lineage. 2 – Column-level lineage.
limit	Yes	number	Maximum results returned per page up to 10,000.
nextID	Yes (from the second call)	string	Use the supplied value to retrieve the next result page.

Request example

```
{
  "ConnectionIds": ["101", "102"],
  "AssetNames": ["storeid", "customer"],
  "ToolNames": ["SQLS", "POWERBI"],
  "ToolTypes": ["DB", "ETL"],
  "AssetPath": "AdventureWorks2014.dbo",
  "AssetType": 2,
  "Limit": 1000,
  "IsFaked": false,
  "Offset": 0,
  "IsDashboard": false,
  "IsMap": false
}
```

Response structure

The response body contains the main items array along with the following fields to control pagination:

- hasMore (boolean) – Indicates whether more result pages are available.
- id (string) – Cursor identifier.
- items (array) – List of asset objects with metadata such as _key, connection details, object type, and timestamps.

Pagination

To retrieve subsequent batches of results, call GET /api/v2.0/assets/query/scroll/{cursorId}. Continue this process until the hasMore field in a response is false.

Response example


```
{
  "items": [
    {
      "_key": "8D1A993B6739DAB65392B54D45F7052E",
      "connectionId": "251",
      "connLogicName": "sqlserverdbwa03testETLTRUE",
      "toolName": "SQLS",
      "toolType": "DB",
      "displayConnectionId": "251",
      "containerObjectName": "sp_built_MrrPerson_MrrCustomer",
      "containerObjectPath": "AdventureWorks2014.dbo",
      "containerObjectType": "SQL_STORED_PROCEDURE",
      "controlFlowPath": "sp_built_MrrPerson_MrrCustomer",
      "controlFlowName": "sp_built_MrrPerson_MrrCustomer",
      "objectID": "ADVENTUREWORKS2014SALESCUSTOMERTABLESTOREID",
      "objectGUID": "8D1A993B6739DAB65392B54D45F7052E",
      "objectType": "TABLE",
      "assetName": "StoreID",
      "dataType": "",
      "precision": "",
      "scale": "",
      "layerName": "Customer",
      "schemaName": "SALES",
      "databaseName": "AdventureWorks2014",
      "tableName": "Customer",
      "isObjectData": true,
      "isMap": false,
      "isVisible": true,
      "updatedDate": "2024-05-26 10:21:51",
      "createDate": "2024-01-01 00:00:00",
      "serverName": "",
      "isSrcColumnOrphan": null
    }
  ],
  "hasMore": false,
  "cursorId": null
}
```


Lineage API

Use the Lineage API to retrieve lineage data that describes the relationships and dependencies between assets.

Key function

Return lineage details for a specific asset.

Endpoint

POST /api/v2.0/lineage

Request body


```
{
  "assetKey": "",
  "depth": 0,
  "direction": 2,
  "limit": 0,
  "assetType": 0
}
```

Request body parameters

Parameter	Required	Type	Description
assetKey	Yes	string	Asset key that defines the lineage starting point
depth	Yes	number	Total number of hops to traverse from the start asset
direction	No	number	The default value is 2. 1 – Input relations only 2 – Output relations only
limit	Yes	number	Total number of assets returned in the lineage graph
assetType	Yes	number	1 – Cross-system lineage 2 – Column-level lineage

Request example

```
{
  "assetKey": "8D1A993B6739DAB65392B54D45F7052E",
  "depth": 12,
  "direction": 1,
  "assetType": 1,
  "limit": 5000,
  "connections": ["106"]
}
```

Response structure

- nodes (array): Asset objects that participate in the lineage graph.
- links (array): Links that connect the nodes.
- mainNode (object): The originating asset for the requested lineage.
- depth and direction: Metadata describing the traversal depth and direction.

Response example

```
"nodes": [
    {
      "Rid": "8D1A993B6739DAB65392B54D45F7052E",
      "name": "StoreID",
      "type": "DB",
      "depth": "0",
      "tool": "SQLS",
      "ObjectGUID": "8D1A993B6739DAB65392B54D45F7052E",
      "isStartNode": false,
      "IsSource": null,
      "in_LINEAGEFLOW": "",
      "out_LINEAGEFLOW": "",10
      "properties": {
        "TYPE": "node",
        "ModuleType": "DATABASE",
        "rid": "8D1A993B6739DAB65392B54D45F7052E",
        "END": "IS_KEY",
        "ToolName": "SQLS",
        "ObjectGUID": "8D1A993B6739DAB65392B54D45F7052E",
        "ContainerToolType": "SQLS",
        "ObjectId": "StoreID",
        "ContainerObjectName": "AdventureWorks2014.dbo",
        "ContainerObjectPath": null,
        "Definition": "",
```

```
        "IsFaked": false,
        "ControlFlowPath": "",
        "ControlFlowName": "",
        "ToolType": "DB",
        "ObjectName": "StoreID",
        "ConnectionName": "AdventureWorks2014",
        "ConnectionID": "106",
        "Project": "AdventureWorks2014",
        "Model": "AdventureWorks2014",
        "Folder": "dbo",
        "ObjectType": "TABLE",
        "Server": "",
        "DB": "AdventureWorks2014",
        "Schema": "dbo",
        "Layer": "StoreID",
        "Table": "StoreID",
        "ViewType": "",
        "in_count": 0,
        "out_count": 1
      }
    },
    {
      "Rid": "F6DC4A4D0BA1C88011F843A31BB6BFD4",
      "name": "CustomerID",
      "type": "DB",
      "depth": "1",
      "tool": "SQLS",
      "ObjectGUID": "F6DC4A4D0BA1C88011F843A31BB6BFD4",
      "isStartNode": false,
      "IsSource": null,
      "in_LINEAGEFLOW": "",
      "out_LINEAGEFLOW": "",
      "properties": {
        "TYPE": "node",
        "ModuleType": "DATABASE",
        "rid": "F6DC4A4D0BA1C88011F843A31BB6BFD4",
        "END": "IS_KEY",
        "ToolName": "SQLS",
        "ObjectGUID": "F6DC4A4D0BA1C88011F843A31BB6BFD4",
        "ContainerToolType": "SQLS",
        "ObjectId": "CustomerID",
        "ContainerObjectName": "AdventureWorks2014.dbo",
        "ContainerObjectPath": null,
        "Definition": "",
        "IsFaked": false,
        "ControlFlowPath": "",
        "ControlFlowName": "",
        "ToolType": "DB",
        "ObjectName": "CustomerID",
        "ConnectionName": "AdventureWorks2014",
        "ConnectionID": "106",
        "Project": "AdventureWorks2014",
        "Model": "AdventureWorks2014",
        "Folder": "dbo",
        "ObjectType": "TABLE",
        "Server": "",
        "DB": "AdventureWorks2014",
        "Schema": "dbo",
        "Layer": "CustomerID",
        "Table": "CustomerID",
        "ViewType": "",
        "in_count": 1,
        "out_count": 0
      }
```

```
    }
  ],
  "links": [
    {
      "FromRid": "8D1A993B6739DAB65392B54D45F7052E",
      "ToRid": "F6DC4A4D0BA1C88011F843A31BB6BFD4",
      "From_ConnectionId": "106",
      "To_ConnectionId": "106",
      "source": 0,
      "target": 1,
      "LinkType": "LINEAGE",
      "Linkid": "LNK-001",
      "LinkDescription": "",
      "LinkedGUID": "LNK-001"
    }
  ],
  "mainNode": {
    "Rid": "8D1A993B6739DAB65392B54D45F7052E",
    "name": "StoreID",
    "type": "DB",
    "depth": "0",
    "tool": "SQLS",
    "ObjectGUID": "8D1A993B6739DAB65392B54D45F7052E",
    "isStartNode": true,
    "IsSource": null,
    "in_LINEAGEFLOW": "",
    "out_LINEAGEFLOW": "",
    "properties": {
      "TYPE": "node",
      "ModuleType": "DATABASE",
      "rid": "8D1A993B6739DAB65392B54D45F7052E",
      "END": "IS_KEY",
      "ToolName": "SQLS",
      "ObjectGUID": "8D1A993B6739DAB65392B54D45F7052E",
      "ContainerToolType": "SQLS",
      "ObjectId": "StoreID",
      "ContainerObjectName": "AdventureWorks2014.dbo",
      "ContainerObjectPath": null,
      "Definition": "",
      "IsFaked": false,
      "ControlFlowPath": "",
      "ControlFlowName": "",
      "ToolType": "DB",
      "ObjectName": "StoreID",
      "ConnectionName": "AdventureWorks2014",
      "ConnectionID": "106",
      "Project": "AdventureWorks2014",
      "Model": "AdventureWorks2014",
      "Folder": "dbo",
      "ObjectType": "TABLE",
      "Server": "",
      "DB": "AdventureWorks2014",
      "Schema": "dbo",
      "Layer": "StoreID",
      "Table": "StoreID",
      "ViewType": "",
      "in_count": 0,
      "out_count": 1
    }
  ],
  "depth": "12"
}
```

Cloudera Data Lineage API: UserEvents documentation for user audit trails

Learn how to use the Cloudera Data Lineage (formerly Octopai) UserEvents API to query user audit events, including login, password reset, and user-management actions recorded automatically by the UserAccount API.

About this task

An audit trail is a security-relevant chronological record that provides documentary evidence of the sequence of activities that have affected a specific operation, procedure, or event. In the context of this API, it records the sequence of user activities or events.

Audit trails are a crucial aspect of security and compliance for many organizations. They are used to detect security incidents, performance issues, and to aid in the recovery from incidents. Additionally, they support the investigation and forensic analysis of how an incident occurred.

The UserEvents functionality contributes to a user audit trail the following ways:

- User authentication: The API logs events related to user authentication, such as successful and failed login attempts. This can help detect potential security risks, like repeated failed login attempts that might indicate a brute-force attack.
- User activity: The API records various user activities like page loads. By tracking these events, administrators can establish a pattern of normal behavior per user, making it easier to identify anomalous actions that could signify a breach.
- Timestamps: Every event logged by the API includes a timestamp. This allows administrators to reconstruct the sequence of events leading up to a particular incident, which is vital in forensic investigations.
- Data source: The IP addresses from which events originate are also recorded. This can be used to identify suspicious activity from unfamiliar sources.

By extracting and analyzing data from the `UserEvents` API, organizations can maintain a comprehensive audit trail that helps uphold security, facilitate incident response, and ensure regulatory compliance.

Events are recorded automatically when users log in, reset passwords, or when an admin creates, updates, or deletes users through the UserAccount API.

Before you begin

- You must be familiar with HTTP methods, specifically POST.
- You must be able to use command-line tools like curl.
- You must have valid Cloudera Data Lineage user credentials that are email and password. To query user events or perform user-management API calls, you must log in with an admin account.

Procedure

1. User login

   To authenticate a user, send a POST request to the following endpoint:

   ```
   https://[***COMPANY NAME***].octopai.com/api/UserAccount/Login
   ```

   Construct and send this request using curl.

   ```
   curl --location 'https://[***COMPANY NAME***].octopai.com/api/UserAccount/Login' \
   --header 'Content-Type: application/json' \
   --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
   --data-raw '{
   "Username": "[***USERNAME***]",
   ```

```
      "Password": "[***PASSWORD***]"
      }'
```

Replace *[***COMPANY NAME***]*, *[***USERNAME***]*, and *[***PASSWORD***]* with your actual values.

Note:

Use an admin account for the steps that follow.

2. Extract access token

   After a successful login, the API returns a JSON response containing an accessToken, which you use for subsequent authenticated requests.

   Successful response example:

```
      {
      "accessToken": "[***ACCESS TOKEN***]",
      "expiration": "2026-07-21T15:36:37Z",
      "userName": "admin@example.com",
      "isAdmin": true,
      "authLevel": "ADMIN"
      }
```

   Extract the accessToken from this response for the next steps.

3. Query user events

   To retrieve user events, send a POST request to the `/api/UserAccount/UserEvents` endpoint. You must authenticate as an admin.

   Include these headers on every request:

   - Authorization: Bearer *[***ACCESS TOKEN***]*
   - Company-Host: *[***COMPANY NAME***]*.octopai.com

   Construct and send the request.

```
      curl --location 'https://[***COMPANY NAME***].octopai.com/api/Use
rAccount/UserEvents' \
      --header 'Content-Type: application/json' \
      --header 'Authorization: Bearer [***ACCESS TOKEN***]' \
      --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
      --data '{
      "PageNumber": "1",
      "RowsInPage": "100",
      "FromDate": "2026-07-21",
      "ToDate": "2026-07-21"
      }'
```

   The request body supports these fields:

   - PageNumber: Page number (1-based).
   - RowsInPage: Number of events per page.
   - FromDate: Start of the date filter. Accepted formats include yyyy-MM-dd, dd-MM-yyyy, and optional HH:mm:ss.
   - ToDate: End of the date filter. Uses the same formats as FromDate.

   If you omit FromDate and ToDate, the API returns events for all dates, subject to pagination. Results are ordered by newest first.

4. Understand the response

   The response from the `/api/UserAccount/UserEvents` endpoint is an array of objects, each representing a user event.

   Successful response example:

```
[
      {
      "type": "DELETE USER SUCCESS",
      "userName": "user@customer.com",
      "source": "203.0.113.42",
      "time": "2026-07-21T12:23:15.047",
      "actor": "admin@customer.com",
      "previousAuthLevel": "ADMIN",
      "newAuthLevel": null
      },
      {
      "type": "ROLE CHANGE SUCCESS",
      "userName": "user@customer.com",
      "source": "203.0.113.42",
      "time": "2026-07-21T12:23:05.94",
      "actor": "admin@customer.com",
      "previousAuthLevel": "VIEWER",
      "newAuthLevel": "ADMIN"
      },
      {
      "type": "LOGIN SUCCESS",
      "userName": "user@customer.com",
      "source": "203.0.113.42",
      "time": "2026-07-21T12:20:10.707",
      "actor": null,
      "previousAuthLevel": null,
      "newAuthLevel": null
      },
      {
      "type": "NEW USER SUCCESS",
      "userName": "user@customer.com",
      "source": "203.0.113.42",
      "time": "2026-07-21T12:19:08.14",
      "actor": "admin@customer.com",
      "previousAuthLevel": null,
      "newAuthLevel": "VIEWER"
      }
      ]
```

   Each event object contains these fields:

   - type: Event type string (for example, LOGIN SUCCESS, NEW USER SUCCESS).
   - userName: Email address of the affected user.
   - source: Client IP address from which the event originated.
   - time: Timestamp of when the event occurred.
   - actor: Email of the admin or authenticated user who performed the action. This field is null when not applicable.
   - previousAuthLevel: User authorization level before a role change or at deletion.
   - newAuthLevel: User authorization level assigned on create or after a role change.

   The API records audit events automatically for login, password reset, user create, user update, and user delete.

   For login, password reset, and other non-user-management events, actor, previousAuthLevel, and newAuthLevel are null.

User-management actions produce these success event types:

- NEW USER SUCCESS: Recorded when an admin creates a user through `POST /api/UserAccount/NewUser`. Populates actor and newAuthLevel.
- DELETE USER SUCCESS: Recorded when an admin deletes a user through `POST /api/UserAccount/DeleteUser`. Populates actor and previousAuthLevel. Delete is a soft delete that sets USER_ACTIVE = NO. A second delete attempt for the same user returns an error and logs DELETE USER FAIL.
- ROLE CHANGE SUCCESS: Recorded only when USER_AUTH_LEVEL changes through `POST /api/UserAccount/UpdateUser`. Populates actor, previousAuthLevel, and newAuthLevel.
- UPDATE USER SUCCESS: Recorded for name or job title updates through `POST /api/UserAccount/UpdateUser`. Does not populate actor, previousAuthLevel, or newAuthLevel.

5. Generate user-management audit events

To produce user-management audit events, perform the corresponding UserAccount API calls with the same Authorization and Company-Host headers, then query `/api/UserAccount/UserEvents` to verify the audit trail.

Create a user:

```
      curl --location 'https://[***COMPANY NAME***].octopai.com/api/UserAccount/NewUser' \
      --header 'Content-Type: application/json' \
      --header 'Authorization: Bearer [***ACCESS TOKEN***]' \
      --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
      --data '{
      "USER_NAME": "[***USER EMAIL***]",
      "USER_FIRST_NAME": "[***FIRST NAME***]",
      "USER_LAST_NAME": "[***LAST NAME***]",
      "USER_AUTH_LEVEL": "VIEWER",
      "USER_PERMISSIONS": "lineage",
      "JOB_TITLE": "[***JOB TITLE***]"
      }'
```

Update a user profile (name only):

```
      curl --location 'https://[***COMPANY NAME***].octopai.com/api/UserAccount/UpdateUser' \
      --header 'Content-Type: application/json' \
      --header 'Authorization: Bearer [***ACCESS TOKEN***]' \
      --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
      --data '{
      "USER_ID": "[***USER ID***]",
      "USER_NAME": "[***USER EMAIL***]",
      "USER_FIRST_NAME": "[***UPDATED FIRST NAME***]",
      "USER_LAST_NAME": "[***LAST NAME***]"
      }'
```

Change a user role:

```
      curl --location 'https://[***COMPANY NAME***].octopai.com/api/UserAccount/UpdateUser' \
      --header 'Content-Type: application/json' \
      --header 'Authorization: Bearer [***ACCESS TOKEN***]' \
      --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
      --data '{
      "USER_ID": "[***USER ID***]",
      "USER_NAME": "[***USER EMAIL***]",
```

```
      "USER_AUTH_LEVEL": "ADMIN"
      }'
```

Delete a user:

```
      curl --location 'https://[***COMPANY NAME***].octopai.com/api/Use
rAccount/DeleteUser' \
      --header 'Content-Type: application/json' \
      --header 'Authorization: Bearer [***ACCESS TOKEN***]' \
      --header 'Company-Host: [***COMPANY NAME***].octopai.com' \
      --data '{
      "USER_ID": "[***USER ID***]",
      "USER_NAME": "[***USER EMAIL***]",
      "USER_FIRST_NAME": "[***UPDATED FIRST NAME***]",
      "USER_LAST_NAME": "[***LAST NAME***]"
      }'
```

What to do next

Store your accessToken securely and refresh it as needed to prevent it from being compromised. Always use HTTPS to make your requests to ensure data security during transit.

Contact Cloudera Support for more details and instructions.