Cloudera Navigator Data Management
# Table of Contents

## About Cloudera Navigator Data Management

- Cloudera Navigator Overview
  - Cloudera Navigator Console
  - Accessing the Cloudera Navigator Console
  - Menu and Icon Overview
  - Displaying Documentation
  - Displaying Version Information
  - Data Stewardship Dashboard
  - Dashboard
  - Data Explorer

## Cloudera Navigator Auditing

- Viewing Audit Analytics
- Using Audit Events to Understand Cluster Activity
- What Did a Specific User Do on a Specific Day?
- Who Deleted Files from Hive Warehouse Directory?
- What Happened to Data in the Database?
- Who ran which operation against a table?
- Exploring Audit Data
- Viewing Audit Events
- Filtering Audit Events
- Cloudera Navigator Audit Event Reports
- Creating Audit Event Reports
- Editing Audit Event Reports
- Downloading Audit Event Reports

## Cloudera Navigator Metadata

- Three Different Classes of Metadata
- Viewing Metadata Analytics
- Defining Properties for Managed Metadata
- Creating Custom Properties with the Cloudera Navigator Console
- Using Cloudera Navigator Console to Manage Properties
- Navigator Built-in Classes
- Defining Metadata with the Navigator API and Navigator SDK
About Cloudera Navigator Data Management

This guide shows you how to use Cloudera Navigator Data Management component for comprehensive data governance, compliance, data stewardship, and other data management tasks.
Cloudera Navigator Overview

Cloudera Navigator Data Management component is a complete solution for data governance, auditing, and related data management tasks that is fully integrated with the Hadoop platform, and that collects governance artifacts—lineage, metadata, and auditing—automatically. Cloudera Navigator lets compliance groups, data stewards, administrators, and others work effectively with data at scale. Working "effectively with data at scale" means that finding specific data entities—to prove to regulators that deposit reserves have not been artificially manipulated, for example—is easier when the data entities in multi-hundred-gigabyte (or terabyte, and beyond) clusters have been cataloged.

An effective catalog of cluster data supports self-service data discovery for an organization. For example, business users can find all the data associated with particular projects by looking for meaningful labels without needing to know the low-level structures within the cluster. Finding all the tables or files relevant for a specific project in a given department can be as simple as applying filters to the Search page. For example, to start exploring the finance team’s data stored in an organization’s Amazon Simple Storage Service (S3), authorized users can select the source type and the tag filters:

Cloudera Navigator enables the cataloging function for Hadoop clusters thanks to its metadata architecture, which lets organizations devise comprehensive metadata models and apply them automatically as data is ingested into the cluster. Using the Cloudera Navigator console, data stewards and other business users can get an at-a-glance view of cluster data through charts, graphs, and histograms that allow further drill-down into all the details, including displaying lineage diagrams that capture transformations to data entities that may have occurred since origination. By tracing data entities back to their source, lineage diagrams show an entity’s provenance and can be used to authenticate values by rendering precisely the transformations that may have occurred.

In addition to its metadata infrastructure, Cloudera Navigator also provides auditing infrastructure. Finding the data needed to pull together reports for regulators using standard Hadoop tools can take days but Cloudera Navigator lets organizations find needed information easily and quickly, from menu selectable reports, dashboards, and the like. Using the analytics feature to look at HDFS utilization, system administrators can glean meaningful information about data operations that might be tying up system resources and many other details. With further drill-down, administrators can identify root causes of issues and can also pro-actively monitor and pre-empt potential issues that could occur due to poor organization or consumption of data resources.

This Cloudera Navigator Data Management guide contains information needed by data stewards, compliance teams, and business users to solve business problems using cluster metadata. The guide also includes information for administrators to configure and manage Cloudera Navigator servers and users. The table lists other documentation for Cloudera Navigator:

<table>
<thead>
<tr>
<th>Introduction</th>
<th>Cloudera Navigator Data Management Overview provides brief overview of common Cloudera Navigator data management functionality for data stewards, governance and compliance teams, data engineers, and administrators. Includes FAQs and an introduction to the Cloudera Navigator APIs with some resources for more information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation</td>
<td>Installing the Cloudera Navigator Data Management Component provides all the details about installing and setting up Cloudera Navigator Data Management component. Cloudera Navigator requires a Cloudera Enterprise license. See Managing Licenses for details.</td>
</tr>
</tbody>
</table>
Upgrade: Upgrading the Cloudera Navigator Data Management Component guides administrators through the upgrade process. Be sure to read the Upgrade Issues and Limitations before starting an upgrade.


Cloudera Navigator Console

The Cloudera Navigator console is the web-based user interface that provides data stewards, compliance groups, auditing teams, data engineers, and other business users access to Cloudera Navigator auditing, metadata, lineage, analytics, and other capabilities.

For example, compliance teams use the Cloudera Navigator console to view audit event reports, save reports for offline use (such as providing documented audit trails to bank auditors), and identify logon attempts by unauthorized users. Data stewards use the Cloudera Navigator console to define managed metadata models that apply to the various use cases in their organizations, define policies that apply the metadata as needed upon ingestion into the cluster, and allow other users to trace the lineage of data that has been transformed over its lifetime in the cluster, back to its source (provenance).

The Cloudera Navigator console provides a unified view of auditing, lineage, and other data management capabilities across all clusters managed by a given Cloudera Manager instance. That is, if the Cloudera Manager instance manages five clusters, the Cloudera Navigator console can gather audit events for all five clusters, as well as manage metadata and display lineage diagrams that allow tracking data sources through any number of transformations to any end result across the five clusters.

Note: The Cloudera Navigator console invokes the REST APIs exposed by the web service on the host supporting the Navigator Metadata Server role instance. The APIs can be called directly by in-house, third-party, and other developers who want to automate tasks or implement functionality not currently provided by the Cloudera Navigator console. See Cloudera Navigator APIs for an overview and other details.

Assuming that the Cloudera Navigator data management component is running, that you have a login account for Cloudera Navigator, and that you know the host name and port of Navigator Metadata Server role (or have been given this information by your system administrator), follow the steps below to access the Cloudera Navigator console.

Accessing the Cloudera Navigator Console

The Cloudera Navigator console can be accessed from the Cloudera Manager Admin Console or directly on the Navigator Metadata Server instance. Only Cloudera Manager users with the role of Navigator Administrator (or Full Administrator) can access Cloudera Navigator console from the Cloudera Manager Admin Console.

From the Cloudera Manager Admin Console:

1. Open your browser.
2. Navigate to the Cloudera Manager Admin Console.
3. Log in as either Navigator Administrator or Full Administrator.
4. From the menu, select Clusters > Cluster-n.
5. Select Cloudera Navigator from the menu.
6. Log in to the Cloudera Navigator console.

To access the Cloudera Navigator console directly:

1. Open your browser.
2. Navigate to the host within the cluster running the Cloudera Navigator Metadata Server role.

   http://fqdn-1.example.com:7187/login.html

   In this example, node 1 of the cluster is running the Navigator Metadata Server role, hosted on the default port 7187. The login page displays.

3. Log in to the Cloudera Navigator console using the credentials assigned by your administrator.

   If your Cloudera Navigator instance has been configured to integrate with your organization's Active Directory (or other LDAP instance) for authentication, your credentials may be simply the same user account and password you use every day for your other production systems.

Menu and Icon Overview

The Cloudera Navigator console opens to an empty Search results page and displays the following menu.

The user_name (the account that is logged in to Cloudera Navigator) and question mark icon (see the table) always display. However, other available menu choices depend on the Cloudera Navigator user role of the logged in account. In this example, the admin account is logged in.

This table provides a reference to all menu choices.

<table>
<thead>
<tr>
<th>Icon or menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>Displays sets of filters that can be applied to find specific entities by source type, type, owner, cluster name, and tags.</td>
</tr>
<tr>
<td>Audits</td>
<td>Displays summary list of current audit events for the selected filter. Define filters for audit events, create reports based on selected time frames and filters, export selected audit details as CSV or JSON files.</td>
</tr>
<tr>
<td>Analytics</td>
<td>Menu for HDFS Analytics and Data Stewardship Analytics selections. This is your entry point for several dashboards that display the overall state of cluster data and usage trends:</td>
</tr>
<tr>
<td></td>
<td>• Data Stewardship Analytics displays Dashboard and Data Explorer tabs.</td>
</tr>
<tr>
<td></td>
<td>• HDFS Analytics displays Metadata and Audit tabs.</td>
</tr>
<tr>
<td>Policies</td>
<td>Displays a list of existing policies and enables you to create new policies.</td>
</tr>
<tr>
<td>Administration</td>
<td>Displays tabs for Managed Metadata, Role Management, and Purge schedule settings.</td>
</tr>
<tr>
<td>user_name</td>
<td>Displays account (admin, for example) currently logged in to Cloudera Navigator. Menu for Command Actions, My Roles, Enable Debug Mode, and Logout selections.</td>
</tr>
<tr>
<td>Help</td>
<td>Help icon and menu. Provides access to Help, API Documentation, and About selections. Select Help to display Cloudera Navigator documentation. Select About to display version information.</td>
</tr>
</tbody>
</table>

Menu selections available for any specific login session depend on the privileges that have been granted to the account holder, through their assigned user role. In other words, a user with the Navigator Administrator user role has all menu selections and functions available, while a user with the Auditing Viewer user role can only access the Audits menu to view, report on, and analyze audit events gathered by the system.

Displaying Documentation

Online documentation is accessible from Cloudera Navigator console as follows:
1. Click Help.
2. Select Help from the menu.

The Cloudera Data Management guide displays.

Note: You can also view the Cloudera Data Management guide outside the scope of a running instance of Cloudera Navigator.

Displaying Version Information

To display the version and build number for the Cloudera Navigator data management component:

1. Click Help.
2. Select About from the menu. The versions for both Cloudera Navigator and the Cloudera Manager instance with which it is associated display, as shown below:

3. Click Close to dismiss the message and redisplay the Cloudera Navigator console.

Data Stewardship Dashboard

The Cloudera Navigator Data Stewardship dashboard captures a variety of information about data, metadata, and user jobs that process the data. The Data Stewardship dashboard provides information and metrics to help you understand the state of the data and data usage, and lets you visualize trends and averages for a variety of data sources and actions.

For example, system administrators use the Dashboard (Analytics > Data Stewardship Dashboard) to look at the Activity Summary page, which provides a comprehensive overall view—databases created, tables created, tables dropped, SQL queries started, and so on.

Administrators can look for trends, the administrator navigates to the Data Explorer tab, makes two selections to filter for the past several weeks on SQL Queries Started as source type and gleans some clues about a rogue SQL query.

Access the dashboard by clicking Analytics and then choosing Data Stewardship on the navigation bar. Specify the source clusters by clicking Source and clicking a cluster name or All Clusters.

On the Data Stewardship dashboard, select a tab for the information you want to view:

- Dashboard. Provides "at-a-glance" information about databases, tables, operations, and files and directories
- Data Explorer. Lets you select cluster actions to view and compare for specific time periods, in conjunction with charts that show averages and trend lines.

The dashboard is divided into the following major information areas:

- Activity Summary
- Databases
Cloudera Navigator Overview

- **Hive Tables**
- **Files and Directories**
- **Operations and Operation Executions**

Dashboard

Activity Summary

<table>
<thead>
<tr>
<th>Activity Summary</th>
<th>Today’s data, compared to previous days</th>
<th>Daily Trend ▼</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Databases Created</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>123</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tables Created</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3K</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tables Altered</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tables Populated</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tables Dropped</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>121</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Partitions Created</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4K</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>HDFS Files Created</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SQL Queries Started</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Accesses Denied</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>138</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Each tile in the Activity Summary section provides summary information for actions on a particular entity type and includes the following:

- The name of the activity
- The number of occurrences for that activity for a time period that you select (daily, weekly, monthly, quarterly, all time)
- A line graph showing activity trends based on the time period that you select

A graphical representation of the time-lapse summary for each activity tile is located at the bottom of the tile. Hovering over a point displays the value for that entity on a particular date. For example, if you select Daily Trend, the number in the graph shows number of occurrences for the day so far (since midnight), and hovering over a graph point shows the number of occurrences for that full day as well as the average for the 20-day period represented by the graph.

The Activity Summary area includes the following information:

<table>
<thead>
<tr>
<th>Entity Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Databases Created</strong></td>
<td>Number of new databases that were added to the cluster.</td>
</tr>
<tr>
<td><strong>Tables Created</strong></td>
<td>Number of new tables that were added to the cluster. Click the value to link to the Search page that shows the search results of the query defined. You can apply filters to narrow the search results and perform any other search actions.</td>
</tr>
<tr>
<td><strong>Tables Altered</strong></td>
<td>Number of tables that were changed.</td>
</tr>
<tr>
<td><strong>Tables Populated</strong></td>
<td>Number of tables that were populated with data.</td>
</tr>
<tr>
<td><strong>Tables Dropped</strong></td>
<td>Number of tables that were deleted.</td>
</tr>
<tr>
<td><strong>Partitions Created</strong></td>
<td>Number of partitions added. You can apply filters to narrow the search results and perform any other search actions.</td>
</tr>
<tr>
<td><strong>HDFS Files Created</strong></td>
<td>Files that were created. Click the new files value to link the Search page that shows the results of the query defined. You can apply filters to narrow the search results and perform any other search actions.</td>
</tr>
</tbody>
</table>
### SQL Queries

<table>
<thead>
<tr>
<th>SQL Queries Started</th>
<th>Number of SQL queries that were run.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Accesses Denied</th>
<th>Number of access attempts by users that were denied.</th>
</tr>
</thead>
</table>

### Databases

#### 114 Databases

<table>
<thead>
<tr>
<th>Database</th>
<th>Table Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>clusterstats</td>
<td>2K</td>
</tr>
<tr>
<td>netsuite</td>
<td>1.2K</td>
</tr>
<tr>
<td>support</td>
<td>1K</td>
</tr>
<tr>
<td>jira</td>
<td>626</td>
</tr>
<tr>
<td>sfdc</td>
<td>434</td>
</tr>
<tr>
<td>warehouse_main</td>
<td>374</td>
</tr>
<tr>
<td>moodle</td>
<td>340</td>
</tr>
<tr>
<td>as_adventure</td>
<td>258</td>
</tr>
<tr>
<td>default</td>
<td>167</td>
</tr>
<tr>
<td>w_house_satellite</td>
<td>141</td>
</tr>
</tbody>
</table>

The Databases area of the Dashboard shows the total number of databases in the source clusters. The top 10 databases, by table count, are displayed in the bar graph.

Click the value next to the Database heading (in this case, 1.5K) to open a Search page showing results from the query "Deleted=Do not show deleted" "Type=Database". Apply or remove filters to refine or broaden the results.
Hover over the bar in the graph to see information about that database, and click the bar to open the Details page for that database. The following figure shows the Details page for the database `nav_policy_db`. 
Hive Tables

8.2K Hive Tables and Views

Top Tables by Partition Count

<table>
<thead>
<tr>
<th>Table</th>
<th>Database</th>
<th>Partition Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 customer_logs</td>
<td>default</td>
<td>120.8K</td>
</tr>
<tr>
<td>2 metrics_production</td>
<td>default</td>
<td>98.1K</td>
</tr>
<tr>
<td>3 usage_scrapes</td>
<td>product_usage</td>
<td>77.7K</td>
</tr>
<tr>
<td>4 metrics_staging</td>
<td>default</td>
<td>23.1K</td>
</tr>
<tr>
<td>5 clusterid_mostrecentcollection...</td>
<td>u_mulyadi</td>
<td>10.4K</td>
</tr>
<tr>
<td>6 opportunity_history</td>
<td>sfdc</td>
<td>7.7K</td>
</tr>
<tr>
<td>7 usage_final</td>
<td>product_usage</td>
<td>7K</td>
</tr>
<tr>
<td>8 distats_parquet</td>
<td>default</td>
<td>6.2K</td>
</tr>
<tr>
<td>9 case_history</td>
<td>sfdc</td>
<td>5.2K</td>
</tr>
<tr>
<td>10 jira_c_history</td>
<td>sfdc</td>
<td>4.9K</td>
</tr>
</tbody>
</table>

The Tables area of the dashboard shows the total number of Hive tables in the cluster. The top 10 tables, by partition count, are displayed in the bar graph.

Click the value next to the Hive Tables heading (in this case, 103.9K) to view matching tables in Search.

Click the bar to open the Details page for that table.

Files and Directories

802 Files 168 Directories

Top by Size

<table>
<thead>
<tr>
<th>File</th>
<th>Path</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 access.log</td>
<td>/dualcore/web_logs</td>
<td>106.3M</td>
</tr>
<tr>
<td>2 ed.data1.txt</td>
<td>/dualcore</td>
<td>31.9M</td>
</tr>
<tr>
<td>3 part-m-00000</td>
<td>/dualcore/ed_data1</td>
<td>26.1M</td>
</tr>
<tr>
<td>4 ed.data2.txt</td>
<td>/dualcore</td>
<td>23.6M</td>
</tr>
<tr>
<td>5 part-r-00000</td>
<td>/dualcore/ed_data2</td>
<td>22.4M</td>
</tr>
<tr>
<td>6 hive-exec.jar</td>
<td>/usr/oozie/share/lib/lib_20170509205421/Hive</td>
<td>10.5M</td>
</tr>
<tr>
<td>7 hive-exec.jar</td>
<td>/usr/oozie/share/lib/lib_20170509205421/spark</td>
<td>10.5M</td>
</tr>
<tr>
<td>8 hive-exec.jar</td>
<td>/usr/oozie/share/lib/lib_20170509205421/spark</td>
<td>10.5M</td>
</tr>
<tr>
<td>9 hive-exec.jar</td>
<td>/usr/oozie/share/lib/lib_20170509205421/spark</td>
<td>10.5M</td>
</tr>
<tr>
<td>10 part-m-00001</td>
<td>/dualcore/orders</td>
<td>15.8M</td>
</tr>
</tbody>
</table>

The Files and Directories area of the Dashboard shows the total number of files and directories in the cluster.

Clicking the value next to the Files or Directory heading (in this case, 64.6M or 16.2M, respectively) to show matching files or directories in Search.

The bar graph displays the top 20 files, based on size. Hover over the bar in the graph to see information about that file. Click the bar to open the Details page for that file.
The Operations and Operation Executions area of the Dashboard shows the total number of operations and operation executions that occurred in the cluster for the specified period of time.

Click the value next to the Operations or Operations Executions for a service to view matching operations or operation executions in Search.

Data Explorer
Cloudera Navigator Auditing

Cloudera Navigator auditing has been designed to meet demanding corporate governance and compliance requirements, especially for organizations in regulated industries. The Navigator Audit Server collects audit events from selected cluster services. Authorized users can use the Cloudera Navigator console to display audit event reports, create reports, and export audit details as CSV or other file formats. They can also search using configurable filters—a specific user name, IP address, or service name, for example—to quickly obtain answers to a variety of questions (see Using Audit Events to Understand Cluster Activity on page 19 for details).

Audit details for HDFS, the distributed storage layer that underlies any Hadoop cluster, are also characterized using histograms and other graphical charts that can be seen in the analytics area of the Cloudera Navigator console.

Events are the activities that occur during regular operations within the cluster. Events often result in messages that record the success or failure of activity along with other relevant details about the specific internal process, such as a hostname, IP address, or service name. Event messages from various services are typically sent to local log files, and from there, the Cloudera Manager Agent process sends them to the Navigator Audit Server (see Navigator Auditing Architecture on page 87 for details).

In addition to individual service events audited by Cloudera Navigator, Cloudera Manager Server records lifecycle events at the cluster, host, role, service, and user level, and records actions that involve licenses and parcels. For example, starting up the cluster is a lifecycle event captured by Cloudera Manager, as is downloading a parcel to update software. Cloudera Manager also captures security-related events, such as adding users, deleting users, login failures, and login successes. For more information about Cloudera Manager auditing capabilities, see Lifecycle and Security Auditing.

Cloudera Navigator tracks and coalesces events collected by Cloudera Manager and generates some of its own events as well. Audit events are viewable in the Cloudera Navigator console. Several pre-configured reports are available from the Audit tab of the Cloudera Navigator console, where you can also filter for specific types of events, create new reports, and export detailed audit reports as CSV or JSON. For example, here is a partial export of Recent Denied Accesses—user accounts without sufficient privileges (user roles) that attempted to log in to Cloudera Navigator but were prevented from doing so:

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Username</th>
<th>IP Address</th>
<th>Service Name</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jun 23, 2017 8:57:01 AM</td>
<td>admin</td>
<td>172.21.2.218</td>
<td>Navigator</td>
<td>authentication</td>
</tr>
<tr>
<td>Jun 23, 2017 8:56:43 AM</td>
<td>rogue-user</td>
<td>172.21.2.218</td>
<td>Navigator</td>
<td>authentication</td>
</tr>
<tr>
<td>Jun 23, 2017 8:56:23 AM</td>
<td>nav-admin</td>
<td>172.21.2.218</td>
<td>Navigator</td>
<td>authentication</td>
</tr>
<tr>
<td>Jun 23, 2017 8:56:11 AM</td>
<td>kelli</td>
<td>172.21.2.218</td>
<td>Navigator</td>
<td>authentication</td>
</tr>
</tbody>
</table>

Viewing Audit Analytics

**Required Role:** Auditing Viewer (or Full Administrator)

1. Log in to the Cloudera Navigator console.
2. Click the Analytics tab and select HDFS from the menu. The Audit tab displays. The Metadata tab may also display if the account has the appropriate permissions. The Source button displays in the banner section of the console if the Cloudera Navigator instance supports multiple clusters.
Click the **Source** button (if it displays, meaning more than one instance) and select an HDFS instance from those available.

3. The Activity tab displays a bar graph that lists the number of files that have been read the number of times listed in the x-axis.

   - To display at the right the directories containing the files that have been read, click an activity bar. This draws a blue selection outline around the bar and selects the Activity checkbox.
   - To select more than one value, grab a bar edge and brush a range of values.
   - To change a range, click a bar, drag to a different range of values, and drop.
   - To reduce a range, grab a bar edge and contract the range.
   - To clear Activity, clear the checkbox. The previous selection is indicated with a gray outline.
   - When you select Activity and the graph had a previous selection, the previous selection is reused. For example, if you had previously selected values spanning six through nine for the number of times files have been read, and you select the checkbox, six through nine will be reselected.

4. In the directory listing on the right:

   - Filter the directories by typing directory strings in the search box and pressing **Enter** or **Return**.
   - **Required Role:** Metadata & Lineage Viewer (or Managed & Custom Metadata Editor, or Full Administrator)

   Add selected directories to a search query and display the Search tab by doing one of the following:
   - Clicking a directory name link.
   - Selecting one or more directory checkboxes and selecting Actions > Show selection in search.

   - **Required Role:** Metadata & Lineage Viewer (or Managed & Custom Metadata Editor, or Full Administrator)

   Add selected directories to the search query of a new policy and display the Policies tab by selecting one or more directory checkboxes and selecting Actions > Create a policy from selection.

   For example, the following screenshot shows files that have been accessed once and match the string `staging`. Each directory has six files that has been accessed.
Using Audit Events to Understand Cluster Activity

Actions performed on the cluster by users and by services are tracked by Cloudera Navigator. Using the filters available on the Audit page of the Cloudera Navigator console lets you quickly find answers to a variety of questions such as:

- **What Did a Specific User Do on a Specific Day?** on page 19
- **Who Deleted Files from Hive Warehouse Directory?** on page 21
- **What Happened to Data in the Database?** on page 22
- **Who ran which operation on a table?**

The examples below step through these common use cases.

To work with Audit Events in the Cloudera Navigator console, you must have an account with the user role of Auditing Viewer or Navigator Administrator.

To view Audit Events in the Cloudera Navigator console and apply filters:

1. Log in to the Cloudera Navigator console.
2. Click the Audits tab. The default view displays a list of all Audit Events for the past hour (note the date range field above the listing, below the Actions button).
3. Configure and apply filters as needed to find specific events by username, by source, for selected time frames, and so on.

**What Did a Specific User Do on a Specific Day?**

A data engineer complains that a job was not running as well as expected one afternoon and wants to find out what might have occurred.

To filter the audit details to these specifics, you need only the user name and the time frame. The example isolates actions by the user name cmjobuser for a one-hour period on November 3, 2016:

1. Limit the list of audit events to those for a specific user by defining a username filter:
   a. Click Filters to open the filter selector.
   b. Select Username from the Select Property... drop-down selector.
c. Set the evaluation operator (=, like, ! =) to = (equals sign).

d. Type the username (cmjobuser for this example).

e. Click **Apply**. The filter is applied to the audit events and the list displays only those events for the selected username.

2. Limit the Audit Events list for the selected username to only those events that occurred on a specific date at a specific time:

a. Click the arrow next to the current date range field to open the date and time range options selector. Time frame options range from the last 2, 6, 12, or 24 hours to the last 7 or last 30 days, but also let you set a custom range.

b. Click the **Custom Range** link to open the date and time range selector.

c. Use the field controls to set the start date and time and the end date and time for your custom time frame, as shown in this example:

d. Click **Apply**.

The date and time range is applied to the Audit Events filtered by username and displays the resulting list in reverse chronological order (end date and time at the top of the list). The Audit Events for username **cmjobuser** during the one-hour period from 3:00 to 4:00 PM on November 3, 2016 are shown below.
Who Deleted Files from Hive Warehouse Directory?

A data steward is wondering what happened to some critical files in the data warehouse—they seem to have been deleted. The organization uses the Hive warehouse located in the default directory:

- /user/hive/warehouse

To find out who deleted files from this directory, create a filter that uses the Source property in conjunction with the Delete operation.

1. Limit the list of Audit Events to only those for the /user/hive/warehouse directory:
   a. Click Filters.
   b. Select Source from the Select Property... drop-down selector.
   c. Set the evaluation operator to like.
   d. Type /user/hive/warehouse in the entry field for the Source property.
   e. Click Apply. The source filter added to the list of filters and applied to Audit Events, returning an updated Audit Events list.

2. Find all delete operations that occurred on the source (the Hive warehouse directory, in this example):
   a. Click Filters. The Source filter displays the filter previously configured. Click the plus sign (+) at the end of the filter definition to add new filter setting fields.
   b. Select Operation from the Select Property... drop-down selector.
   c. Set the evaluation operator to =.
   d. Type delete in the entry field for Operation property.
   e. Click Apply. The filter to select delete operations is added to the list of filters and applied to Audit Events. An updated Audit Events list is returned, showing only delete operations for sources similar to /user/hive/warehouse.

The results show both completed and attempted delete operations (attempted but unsuccessful are in red). All these events are associated with username navigator_user from the same IP address and using the same service (hdfs) over the course of a day (June 9, 2016).
With this information, a Navigator Administrator can go to the Administration tab of the Cloudera Navigator console and find out who is associated with the navigator_user account.

- Log in to the Cloudera Navigator console.
- Click the Administration tab.
- Click the Role Management tab.

What Happened to Data in the Database?

The data files associated with a database are typically partitioned into files or directories by date, often by year. The system admin discovers that the data from 2015 is missing from the production database—only data prior to 2010 has been archived off the system, so data from 2015 should still be there.

Files for data created in 2015 include 2015 in their filenames.

To determine what happened to the data stored in folders and files in the year 2015, do the following:

1. Filter the list of events for sources containing 2015:
   a. Click Filters.
   b. Select Source from the Select Property... drop-down selector. Sources can be paths to HDFS files or directories.
   c. Set the evaluation operator to like.
   d. Type 2015 in the entry field for the Source property.
   e. Click Apply. The source filter added to the list of filters and applied to Audit Events, returning an updated Audit Events list.

2. Filter the list of events for the delete operation:
   a. Click Filters. The Source filter displays the filter previously configured. Click the plus sign (+) at the end of the filter definition to add new filter setting fields.
   b. Select Operation from the Select Property... drop-down selector.
   c. Set the evaluation operator to =.
   d. Type delete in the entry field for the Operation property.
   e. Click Apply. The operation filter added to the list of filters and applied to Audit Events, returning an updated list containing Audit Events for delete operations from sources like 2015.

3. Set the date range to one year:
   a. Click the date-time field at the top right of the Audit Events page.
   b. To set the range to be the last year, click Custom Range. The Selected Range field is enabled for input.
c. In the left date field, use the field controls to specify a date one year ago.

d. Click Apply.

The result of all filters and the selected time frame reveals that the hdfs username deleted directories containing 2015 in their names:

Who ran which operation against a table?

When a change occurs to a data asset, it can cause problems to reports or other processes that use the data asset. You can use the Audit events to find when and what caused a change in a data asset. This example describes how to use audits to determine what caused a schema change to a table so you can identify the user or process that may be causing unwanted changes.

To find all operations against a specific table in a general time range:

1. Filter the list of events for the table.
   a. Click Filters.
   b. Select Table Name from the Select Property... drop-down selector.
   c. Set the evaluation operator to like.
   d. Type the part or all of the table name in the entry field for the Table Name property.
   e. Click Apply. The source filter added to the list of filters and applied to Audit Events, returning an updated Audit Events list.

2. Set the date range to the general time frame where the change occurred:
   a. Click the date-time field at the top right of the Audit Events page.
   b. To set the range, click Custom Range. The Selected Range field is enabled for input.
   c. In the left date field, use the field controls to specify the earliest date where the change may have occurred.
      To include the entire day, set the time to indicate the beginning of the day, such as 12:01 AM.
   d. In the right date field, specify the latest date where the change may have occurred.
      To include the entire day, set the time to indicate the end of the day, such as 11:59 PM.
   e. Click Apply.

3. Review the results, looking at both the operation and the service name.

4. If the results are too large, consider adding additional filters:
   - By Service Name. For example, to show all Hive operations, use Service Name like hive.
   - By Operation. For example, to show only Hive operations that change the table metadata, use Operation like alter.
Exploring Audit Data

Required Role: Auditing Viewer (or Full Administrator)

Viewing Audit Events

1. Click the Audits tab. By default, the Audit Events report opens, listing all events that occurred within the last hour, with the most recent at the top:

   The Audit Events and Recent Denied Accesses reports are available by default. You create your own reports and apply a variety of filters as detailed in the next section.

Filtering Audit Events

You filter audit events by specifying a time range or adding one or more filters containing an audit event field, operator, and value.

Specifying a Time Range

1. Click the date-time range at the top right of the Audits tab to open a range selector.
2. Select one of the pre-defined ranges.
   To enter a specific time range, choose **Custom range** and use the controls in **Selected Range** to specify start and end dates and times.

3. Click **Apply**.

Adding a Filter

1. Click the **Filters** link.

   The Filters pane displays.

2. Click [+] if a filter field is not already open.

3. Choose a field in the **Select Property** drop-down list.
   The properties correspond to the audit fields listed in **Service Audit Events** on page 168.

4. Choose an operator in the operator drop-down list.

   Choose from **=** (equals), **!=** (not equals), **like**, and **not like**. The **like** operator matches to substrings. For example, to see all the audit events for files created in the folder `/user/joe/out`, specify **Source like /user/joe/out**.

   Use **not like** to filter out events based on a partial string match. For example, to show audit events for a set of IP addresses that start with the same sequence 172.39.109, you can specify **IP Address not like 172.39.109**.

5. Type a field value in the value text field.
   For example, to see the events listed for Navigator, enter **Service Name = Navigator**.

6. To specify an additional filter, click [+] and define the new criteria.
   Audit events show that match all filters (AND).

7. Click **Apply**.
A filter breadcrumb appears above the list of audit events and the list of events displays all events that match all the filter criteria.

Removing a Filter

1. Click the x next to the filter breadcrumb.

You can also click Filters and modify or remove filter criteria.

Cloudera Navigator Audit Event Reports

**Required Role:** Auditing Viewer (or Full Administrator)

Cloudera Navigator console provides two default reports for display by default:

- Audit Events listing, which displays all recent events in reverse chronological order
- Recent Denied Accesses, which displays all accesses within the cluster and highlights denied access attempts in red, for easy identification.

These two default reports are starting points. Use the Audit Report and define filters to create new reports can be created by defining and applying filters to fine-tune the selected results. Give the filtered report specifications for future use, and export (download) any report to CSV and JSON file formats.

In addition, the metadata about the audit reports you create and save is recorded by the Navigator Metadata Server. This section shows you how to use the Cloudera Navigator console to create audit reports.

**Note:** All steps below start from the Cloudera Navigator console.

Creating Audit Event Reports

Selecting the Audit menu in the Cloudera Navigator console displays the Audit Events report. This report displays all audit events captured in the last 1 hour. You can modify the filters configured for this report and save it, giving it a new name, as follows.

1. To save a filtered version of the Audit Events report:
   a. Optionally specify filters (see Filtering Audit Events on page 24).
   b. Click Save As Report.
      - Create a new report by clicking New Report.
2. Enter a report name.
3. In the **Default time range** field, specify a relative time range. If you had specified a custom absolute time range before selecting **Save As Report**, the *custom absolute time range is discarded.*
4. Optionally add filters (see Filtering Audit Events on page 24).
5. Click **Save**.

### Editing Audit Event Reports

1. In the left pane, click a report name.
2. Click **Edit Report**.
3. In the **Default time range** field, specify a relative time range. If you had specified a custom absolute time range before selecting **Save As Report**, the *custom absolute time range is discarded.*
4. Optionally add filters (see Filtering Audit Events on page 24).
5. Click **Save**.

### Downloading Audit Event Reports

You can download audit event reports in the Cloudera Navigator console or using in CSV and JSON formats. An audit event contains the following fields:

- timestamp
- service
- username
- ipAddress
- command
- resource
- allowed
- [operationText]
- serviceValues

The values of the **resource** and **serviceValues** fields depend on the type of the service. In addition, Hive, Hue, Impala, and Sentry events have the **operationText** field, which contains the operation string. See Service Audit Events on page 168 for details.

In addition to downloading audit events, you can configure the Navigator Audit Server to publish audit events to a Kafka topic or syslog. See Publishing Audit Events.

### Downloading Audit Event Reports from

1. Do one of the following:
   - Add filters (see Filtering Audit Events on page 24).
   - In the left pane, click a report name.
2. Select **Export > format**, where **format** is CSV or JSON.

### Downloading Audit Events Using the Audit API

You can filter and download audit events using the Cloudera Navigator APIs.

#### Hive Audit Events Using the Audit API

To use the API to download the audits events for a service named hive, use the **audits** endpoint with the GET method:

```
http://fqdn-n.example.com:port/api/APIversion/audits/?parameters
```

where **fqdn-n.example.com** is the host running the Navigator Metadata Server role instance listening for HTTP connections at the specified **port** number (7187 is the default port number). **APIversion** is the running version of the
API as indicated in the footer of the API documentation (available from the Help menu in the Navigator console) or by calling `http://fqdn-n.example.com:port/api/version`.

For example:

```
curl http://node1.example.com:7187/api/v13/audits/?query=service%3D%3Dhive&startTime=1431025200000&endTime=1431032400000&limit=5&offset=0&format=JSON&attachment=false -X GET -u username:password
```

The `startTime` and `endTime` parameters are required and must be specified in **epoch time** in milliseconds.

The request could return the following JSON items:

```json
[{
  "timestamp": "2015-05-07T20:34:39.923Z",
  "service": "hive",
  "username": "hdfs",
  "ipAddress": "12.20.199.170",
  "command": "QUERY",
  "resource": "default:sample_08",
  "operationText": "INSERT OVERWRITE \n  TABLE sample_09 \nSELECT \n  sample_07.code,sample_08.description \n  FROM sample_07 \n  JOIN sample_08 \n  WHERE sample_08.code = sample_07.code",
  "allowed": true,
  "serviceValues": {
    "object_type": "TABLE",
    "database_name": "default",
    "operation_text": "INSERT OVERWRITE \n  TABLE sample_09 \nSELECT \n  sample_07.code,sample_08.description \n  FROM sample_07 \n  JOIN sample_08 \n  WHERE sample_08.code = sample_07.code",
    "resource_path": "/user/hive/warehouse/sample_08",
    "table_name": "sample_08"
  }
}, {
  "timestamp": "2015-05-07T20:33:50.287Z",
  "service": "hive",
  "username": "hdfs",
  "ipAddress": "12.20.199.170",
  "command": "SWITCHDATABASE",
  "resource": "default:"
  "operationText": "USE default",
  "allowed": true,
  "serviceValues": {
    "object_type": "DATABASE",
    "database_name": "default",
    "operation_text": "USE default",
    "resource_path": "/user/hive/warehouse",
    "table_name": ""
  }
}, {
  "timestamp": "2015-05-07T20:33:23.792Z",
  "service": "hive",
  "username": "hdfs",
  "ipAddress": "12.20.199.170",
  "command": "CREATETABLE",
  "resource": "default:"
  "operationText": "CREATE TABLE sample_09 (code string,description string) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '\t' STORED AS TextFile",
  "allowed": true,
  "serviceValues": {
    "object_type": "DATABASE",
    "database_name": "default",
    "operation_text": "CREATE TABLE sample_09 (code string,description string) ROW
FORMAT
DELIMITED FIELDS TERMINATED BY '\t' STORED AS TextFile",
    "resource_path": "/user/hive/warehouse",
    "table_name": ""
  }
}
]
Cloudera Navigator Metadata

Cloudera Navigator lets organizations catalog data contained in Hadoop clusters. With data entities in the cluster tagged with relevant metadata properties, data stewards can provide curated datasets, business users can do their own self-service discovery, and system administrators can develop effective archival strategies.

The Navigator Metadata Server is the role instance that provides the metadata definition, tagging, and management system. Any given entity can be identified by one or more of the three different classes of metadata.

Cloudera Navigator console lets you view metadata through various dashboards, such as the Data Stewardship Dashboard and Data Explorer, that provide at-a-glance views of cluster assets.

This section focuses on the HDFS Analytics menu of the Cloudera Navigator console.

Three Different Classes of Metadata

Cloudera Navigator supports extraction of three different types of metadata from the data contained in the cluster. The characteristics of each type are summarized in the table.

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Usage Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Metadata</td>
<td>Characteristics inherent to the entity that are obtained when extracted.</td>
<td>Not modifiable.</td>
</tr>
<tr>
<td>Managed Metadata</td>
<td>Descriptions, tags, and key-value pairs that can be added to entities after extraction. Keys are defined within namespaces, and values can be constrained by type (Text, Number, Boolean, Date, Enumeration, for example).</td>
<td>Add to entities or modify after extraction only.</td>
</tr>
<tr>
<td>Custom Metadata</td>
<td>Key-value pairs that can be added to entities before or after extraction. Displayed in the Tags area of the Details page for a given entity.</td>
<td>Add to entities before or after extraction.</td>
</tr>
</tbody>
</table>

The screenshot below (from the Cloudera Navigator console) shows the details of a web log saved to HDFS as a comma-separated value (CSV) file. This particular entity has all three types of metadata associated with it:

- The Technical Metadata was provided to the entity by the source system, in this example, HDFS.
- The Managed Metadata was defined by a data steward from the Finance department to augment entities processed by the system with properties that enable self-service data discovery for cluster data. That is, business users looking for data handled by the Finance department can more easily locate files that have been labeled with Managed Metadata.
- Custom Metadata has also been applied to this file (see the Tags area of file details below).
Technical Metadata is obtained from the source entity and cannot be modified. Common examples of Technical Metadata include an entity's name, type (directory or file, for example), path, creation date and time, and access permissions. For entities created or managed by cluster services, Technical Metadata may include the name of the service that manages or uses that entity and relations—parent-child, data flow, and instance of—between entities.

As another example, Technical Metadata for an Amazon S3 bucket includes Bucket name, Region (AWS Region, such as us-west-1), S3 Encryption, S3 Storage Class, S3 Etag, Source (S3), and so on. Technical Metadata is simply whatever metadata is provided for the entity by the system that created the entity.

For example, for Hive entities, Cloudera Navigator extracts the extended attributes added by Hive clients to the entity.

Viewing Metadata Analytics

Required Role: Metadata & Lineage Viewer and Policy Editor (or Full Administrator)

1. Open your browser.
2. Navigate to the host within the cluster running the Navigator Metadata Server role as shown in this example (7187 is the default port for Navigator Metadata Server):

   http://fqdn-1.example.com:7187/login.html

   The login page displays.
3. Log in to the Cloudera Navigator console using the credentials assigned by your administrator.
4. Click the Analytics tab. The Metadata analytics tab displays.
5. Click the Source button and select an HDFS service instance from the drop-down list.
6. The Metadata tab displays a set of bar graphs that list the number of files that satisfy groups of values for last access time, created time, size, block size, and replication count.

- To display the files at the right, click a bar. This draws a blue selection outline around the bar and selects the property checkbox.
- To select more than one value, grab a bar edge and brush a range of values.
- To change a range, click a bar, drag to a different range of values, and then drop.
- To reduce a range, grab a bar edge and contract the range.
- To clear a property, clear the checkbox. The previous selection is indicated with a gray outline.
- When you select a previously selected property, the previous selection is reused. For example, if you had previously selected one and three for replication count, and you reselect the replication count checkbox, the values one and three are reselected.
- To clear all current and past selections, click **Clear all selections**.

7. In the listing on the right, select an option to display the number of files by directory, owner, or tag. In the listing:

- Filter the selections by typing strings in the search box and pressing **Enter** or **Return**.
- Add categories (directory, owner, or tag) to a search query and display the Search tab by doing one of the following:
  - Clicking a directory, owner, or tag name link.
  - Selecting **Actions > Show in search**. To further refine the query, select one or more checkboxes, and select **Actions > Show selection in search**.

**Required Role:** **Policy Editor** (or Full Administrator)

Add categories to the search query of a new policy and display the Policies tab by selecting **Actions > Create a policy**. To further refine the query, select one or more checkboxes, and select **Actions > Create a policy from selection**.

---

**Defining Properties for Managed Metadata**

**Required Role:** **Metadata Administrator** (or Full Administrator)

You can use managed metadata to add typed metadata to classes of entities. You can add namespaces and properties.

A **namespace** is a container for properties. Four namespaces are reserved:

- `nav` for Navigator metadata classes (for example, `fselement` and user-defined custom fields)
- `up` (custom metadata)
- `tp` (technical properties)
- `xt` (partner applications)

The combination of namespace and property name must be unique.

**Note:** The Cloudera Navigator console cannot delete namespaces. Empty namespaces can be deleted using the Cloudera Navigator APIs.

A property can be one of the following types:

- Boolean
- date
- integer
- long
- float
- double
- text (with optional maximum length and regular expression validation criteria)
A property can be single-valued or assume multiple values.

After metadata properties have been assigned to specific entities in the cluster, the property types and values can be used as filters for Cloudera Navigator Search, to find specific entities.

Creating Custom Properties with the Cloudera Navigator Console

To create custom properties:

1. **Log in to the Cloudera Navigator console.**
2. Click the Administration link in the upper right. The Managed Metadata tab displays the list of namespaces and the properties defined in the namespaces.
3. Click the New Property... button.
4. In the Classes field, click the filter icon ( ) or start typing one of the Cloudera Navigator entity built-in class names.
5. Select the class of entities to which the property applies. To clear the field, hover over the field and click the delete icon (x) that displays at the right of the field.
6. Click the Namespace field and select a namespace. If the Namespace drop-down list is empty, click Create Namespace....
   a. Specify a namespace name and optional description.
   b. Click Continue.
7. Add the name for the property.
   The name can contain letters, numbers, underscores, and hyphens and can be up to 50 characters long.
8. Specify an optional description.
9. Select the Multivalued Enable checkbox if the property can have more than one value. For example, an emailFrom property should accept only one value, but an emailTo property could accept more than one value.
10. In the Type drop-down list, select the property type and specify constraints on the value.
    - **Boolean** - Boolean: true or false.
    - **Date** - Date and time.
    - **Enumeration** - A set of values. In the Enumeration field, type valid enumeration values and press Enter or Tab.
    - **Number** - A number. In the Number Type field, select the type of the number: Integer, Long, Float, Double.
    - **Text** - A string.
      - **Maximum Length** - The maximum length of the string.
      - **Regular Expression** - A regular expression that determines whether a string is a valid value. You can test the expression by clicking Show regex tester and entering input that you expect to match the expression.

If you change the regular expression, click outside the field to trigger the tester to re-evaluate the expression.

11. Click Continue to Review. The Review screen displays.
12. Click **Create** to create the property, **Cancel** to return to the Properties page, or **Back to Edit Property** to continue editing the property.

**Example Properties**

The following figure shows two properties in the namespace `MailAnnotation` that apply to entities of the `HDFS Entity` class (HDFS files and directories). The `emailFrom` property is of type `TEXT` and can be assigned a single value. The `MailTo` property is also of type `TEXT` but can have multiple values.

**Properties**

See the [documentation](#) before creating or modifying properties.

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Type</th>
<th>Classes</th>
<th>Multivalued</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>MailAnnotation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EmailFrom</td>
<td>Active</td>
<td>Text</td>
<td>HDFS Entity</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>EmailTo</td>
<td>Active</td>
<td>Text</td>
<td>HDFS Entity</td>
<td>Yes</td>
<td>Actions</td>
</tr>
</tbody>
</table>

**Using Cloudera Navigator Console to Manage Properties**

You can view managed metadata property summary details by clicking property name in the Properties table, or by clicking the **Actions** box in the property row and then clicking **View** in the dropdown.

You can also edit some aspects of a property, delete and restore a property, and purge a deleted property.

**Editing a Property**

After a property is created, you can edit property data in the following ways:

- Add **classes** to which the property applies
- Add and remove enumeration values
- Change the description
- Change the maximum length
- Change the regex pattern

1. Log in to the Cloudera Navigator console using administrator credentials roles:
   - Cloudera Manager Full Administrator
   - Cloudera Manager Navigator Administrator
   - Cloudera Navigator Full Administrator
   - Cloudera Navigator Metadata Administrator

2. Click the **Administration** link in the upper right. The **Managed Metadata** tab displays the list of namespaces and the properties defined in the namespaces.

3. Open the property Edit page by clicking the **Actions** box in the property row and then clicking **Edit** in the dropdown.

4. In the Additional Class field, click the **or** type the name of a Cloudera Navigator **class** entity. For example, start typing "Hive.." to see

5. Select the class of entities to which the property applies. To clear the field, hover over the field and click the delete icon (x) that displays at the right of the field.

6. In the Description field, add a description or edit an existing description.

7. If the property is of the Enumeration type, you can add or remove values in the Enumeration field.
Note: If you delete an enumeration value, it cannot be used in new Managed Metadata assignments, but it continues to exist in entities or policies that are already using them.

8. For Text properties:
   - In the Maximum Length field, add or change the value for the maximum length.
   - In the Regular Expression field, edit the expression. Click Show regex tester to test input against any changes you make.

9. Click Continue to Review. The Review screen displays.
10. Click Update to commit the change or Back to Edit Property to continue editing the property, or Cancel to return to the Properties page.

Deleting, Restoring, and Purging Managed Metadata Properties

After a property is deleted, it cannot be assigned to entities. However, the property still exists and the entities that have already been tagged with the property retain the tag until the purge operation is run.

Deleted properties display status as Deleted in the Cloudera Navigator console (on the Managed Metadata tab of the Administration menu). For example, the EmailFrom property has been deleted:

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Type</th>
<th>Classes</th>
<th>Multivalued</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>MailAnnotation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EmailFrom</td>
<td>Deleted</td>
<td>Text</td>
<td>HDFS Entity</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>EmailTo</td>
<td>Active</td>
<td>Text</td>
<td>HDFS Entity</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

The Status displays only to users with the Navigator Administrator or Managed & Custom Metadata Editor user roles.

The Cloudera Navigator purge process permanently removes properties and any values from all entities. Policies that assign metadata using a property that has been purged will fail the next time they are run. Because deleted properties are not removed from the system until they have been purged, the name of any deleted property cannot be re-used until after purging the system.

Deleting a Property

1. In the Properties table, for the property that you are deleting, click the Actions button, and then click Delete in the drop-down menu.
2. In the Delete Property dialog box, review the property deletion information. If any entities are affected, you see a View affected entities link; click to see all entities that use the property.
3. Click Confirm Delete to delete the property, or click Cancel.

Restoring a Property

If you have not yet purged a deleted property, you can restore it.

- In the Properties table, for the property that you are restoring, click the Actions button, and then click Restore in the drop-down menu.

Purging a Property

You can permanently remove deleted properties by purging them. All values assigned to the deleted properties are lost; however, the affected entities are not deleted. Purging permanently removes all properties marked as Deleted in the Status column.
Note: Purging is a long-running task and may take some time. Navigator is unavailable to all users until the purge is completed.

1. In the Properties table, click **Purge Deleted Properties**. The Purge all Deleted Properties dialog box opens, describing the effects of the purge and reporting the number of entities that use the property.

2. In the Purge all Deleted Properties dialog box, click **Confirm Purge** to permanently remove all deleted properties, or click **Cancel** to return to the Properties page.

### Navigator Built-in Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS Dataset</td>
<td>Logical dataset backed by a path in HDFS.</td>
</tr>
<tr>
<td>HDFS Dataset Field</td>
<td>Field in an HDFS dataset.</td>
</tr>
<tr>
<td>HDFS Entity</td>
<td>HDFS file or directory.</td>
</tr>
<tr>
<td>Hive Column</td>
<td>Column in a Hive table.</td>
</tr>
<tr>
<td>Hive Database</td>
<td>Hive database.</td>
</tr>
<tr>
<td>Hive Partition</td>
<td>Partition of a Hive table.</td>
</tr>
<tr>
<td>Hive Query</td>
<td>Hive query template.</td>
</tr>
<tr>
<td>Hive Query Execution</td>
<td>Instance of a Hive query.</td>
</tr>
<tr>
<td>Hive Query Part</td>
<td>Component of a Hive query that maps specific input columns to output columns.</td>
</tr>
<tr>
<td>Hive Table</td>
<td>A Hive table.</td>
</tr>
<tr>
<td>Hive View</td>
<td>View on one or more Hive tables.</td>
</tr>
<tr>
<td>Impala Query</td>
<td>Impala query template.</td>
</tr>
<tr>
<td>Impala Query Execution</td>
<td>Instance of an Impala query.</td>
</tr>
<tr>
<td>Impala Query Part</td>
<td>Component of an Impala query that maps specific input columns to output columns.</td>
</tr>
<tr>
<td>Job Instance</td>
<td>Instance of a MapReduce, YARN, or Spark job.</td>
</tr>
<tr>
<td>Job Template</td>
<td>Template for a MapReduce, YARN, or Spark job.</td>
</tr>
<tr>
<td>Oozie Workflow</td>
<td>Template for an Oozie workflow.</td>
</tr>
<tr>
<td>Oozie Workflow Instance</td>
<td>Instance of an Oozie workflow.</td>
</tr>
<tr>
<td>Pig Field</td>
<td>Field for a relation in Pig; similar to a column in a Hive table.</td>
</tr>
<tr>
<td>Pig Operation</td>
<td>Template for a Pig transformation.</td>
</tr>
<tr>
<td>Pig Operation Execution</td>
<td>Instance of a Pig transformation.</td>
</tr>
<tr>
<td>Pig Relation</td>
<td>Pig relation; similar to a Hive table.</td>
</tr>
<tr>
<td>S3 Bucket</td>
<td>A bucket in S3.</td>
</tr>
<tr>
<td>S3 Object</td>
<td>A file or directory in an S3 bucket.</td>
</tr>
<tr>
<td>Sqoop Export Sub-operation</td>
<td>Sqoop export component that connects specific columns.</td>
</tr>
<tr>
<td>Sqoop Import Query</td>
<td>Sqoop import job with query options.</td>
</tr>
</tbody>
</table>
### Defining Metadata with the Navigator API and Navigator SDK

In addition to accessing and defining metadata with the Cloudera Navigator console, you can also use the Cloudera Navigator API and the Navigator SDK.

For information on the Navigator API, see Cloudera Navigator APIs on page 146.

The Navigator SDK is a client library that can be used to extract metadata directly from Navigator Metadata Server or to enrich metadata with custom metadata models, entities, and relationships. The SDK is located in Github in the cloudera/navigator-sdk repository.

### Adding and Editing Metadata

**Required Role:** Metadata Administrator (or Full Administrator)

Cloudera Navigator provides two different types of customer-defined metadata that can be applied to entities extracted from the cluster:

- **Managed metadata** consists of the comprehensive organization-wide labeling scheme typically set up by data stewards to provide centralized, curated data sets. Different namespaces can be created for different departments, functional groups, or data type, or any categorization scheme needed, and each namespace contains the key-value pairs that are used to tag entities. Values can be constrained by data type (text, number, boolean, date, enumeration, for example). Managed metadata can be applied to entities after extraction (not before), and can be applied automatically by using the API or configuring the Navigator Metadata Server to apply policies.

- **Custom metadata** consists of key-value pairs that users define on an ad hoc basis and can be applied to entities before and after extraction. Business users who want to classify and organize data for their own use can use the Cloudera Navigator console to do so easily, as detailed below.

#### Editing Metadata Using the Cloudera Navigator Console

1. Log in to the Cloudera Navigator console.
2. **Search** for any entity in the Cloudera Navigator console.
3. Click an entity link returned in the search. The Details tab displays.
4. To the left of the Details tab, click **Actions > Edit Metadata**. The Edit Metadata dialog box drops down.
5. Add metadata fields:
   - In the Name field, type a new display name.
   - In the Description field, type a description (500 characters maximum).
   - **Managed Metadata**
     1. Click the ▼ and select a property.
     2. Click the value field after the : to display type-specific selection controls such as integer spinners and date selection controls. Either type the value or use the controls to select a value.

   Click the plus icon (+) to add another managed property key-value pair or another value for a given key.

---

**Class** | **Description**
---|---
Sqoop Import Sub-operation | Sqoop import component that connects specific columns.
Sqoop Operation Execution | Instance of a Sqoop job.
Sqoop Table Export | Sqoop table export operation template.
Sqoop Table Import | Sqoop table import operation template.
User Sub-operation | User-specified sub-operation of a MapReduce or YARN job; used for specifying custom column-level lineage.
• **Custom Metadata**: In the Tags field, type a tag and press **Enter** or **Tab** to create new tag entries.

• **Key-Value Pairs**

  1. Click the plus icon to add a key-value pair.
  2. Type a key and a value. You can specify special characters (for example, ",", ",") in the name, but it makes searching for the entity more difficult because some characters collide with special characters in the **search syntax**.

**Note**: The Key-Value Pairs field is used for specifying custom metadata only. Managed metadata includes a namespace property which cannot be entered in this field.

6. Click **Save**. The newly defined metadata appears in the Managed Metadata or Custom Metadata pane.

**Custom Metadata Example**

In the following example, the tag **archive_personal** and the property **year** with value **2015** have been added to the file **2015_11_20**:

After you save, the metadata appears in the Tags and Custom Metadata panes:
Managed Metadata Example

The following example shows the Department and RetainUntil managed properties for the customers entity:

After you specify the values and save, the properties display in the Managed Metadata pane:

Finding Specific Entities by Searching Metadata

Required Role: Metadata Viewer (or Metadata Administrator, Full Administrator)

You can find entities matching selected criteria using the Cloudera Navigator console, as detailed below.

Searching Metadata

1. Open your browser.
2. Navigate to the host within the cluster running the Cloudera Navigator Metadata Server role.

In this example, node 1 of the cluster is running the Navigator Metadata Server role, hosted on the default port 7187. The login page displays.
3. Log in to the Cloudera Navigator console using the credentials assigned by your administrator. The Cloudera Navigator console opens to the Search tab.

4. To display all entity types in the search results, click Clear all filters. Or, limit search results to specific source types, entity types, or other characteristics by applying filters.

5. Enter the search strings in the Search box.

Filter Example

The following filter example demonstrates how to narrow search results by selecting a built-in Source Type filter set to HDFS and the technical property PERMISSIONS with the value rwxrwxrwx. Breadcrumbs for each filter value show above the results. The "Deleted = Do not show Deleted" breadcrumb is included in the filter by default.

When you select a property value, a filter breadcrumb shows above the search results. Click the "x" in the breadcrumb to remove that criteria from the filter.

You can also display the details of the search query by choosing Actions Show full query.
Viewing Search Results

The Search Results pane displays the number of matching entries in pages listing 25 entities per page. You can view the pages using the page control at the bottom of each page.

Each entry in the result list contains:

- Source type
- Name - A link to a page that displays the entity details and lineage diagram
- Properties
- If Hue is running, a link at the far right labeled View in Hue that opens the Hue browser for the entity. For example:

```
Hive Table
web_logs
/default
Automated content collection from US sites, accumulated over all corporate bran...
Created Dec 27, 2017 2:47 AM Owner admin Source HIVE-1
```

Displaying Entity Details

The entity Details page displays all three types of metadata for an entity—technical, managed, and custom—and entity type-specific information:

- HDFS directories - Directory contents
- HDFS datasets and fields - Schema
- Hive and Impala databases - Tables and views
- Hive tables - Extended attributes, table schema, partitions
- Impala tables - Table schema
- MapReduce, YARN, and Spark operations - Instances
- Pig operation executions - Tables
- S3 buckets and folders - Folder contents

All entity types can display inputs and outputs. See Enabling Inputs and Outputs to Display.

If managed properties have been defined for a particular entity type, the Show All checkbox in the Managed Metadata pane displays all properties that can be assigned values for the selected entity. To display only those properties that have values, clear the checkbox. If all properties have values, the checkbox has no effect.
To display entity details:

1. Perform a search.
2. In the search results, click an entity name link. The Details tab displays.

**Hive Table Entity Details**

For example, if you click the Hive table `web_logs` link in the search result displayed in preceding example, you see the following details:

The caret > indicates fields with content; click to expand them. In addition to managed metadata, custom metadata, and technical metadata, you can view the names of the columns and the inputs and outputs.

**Managed Metadata Example**

The following account table entity has two managed properties in the `realestate` namespace: Department and AccessAttempts.

The caret > indicates fields with content; click to expand them. In addition to managed metadata, custom metadata, and technical metadata, you can view the names of the columns and the inputs and outputs.
Filtering Search Results

To filter search results, choose filter values in the Filters pane or type search strings in the Search box.

The Filters pane lists default properties (source type, type, owner, cluster, and tags) and their values. You can add or remove properties from this list: click Add Filters... and choose from the defined technical and metadata properties. To remove non-default filter properties, click the x in the filter.

Note that when filter values are timestamps, the server stores the timestamp in UTC, and the UI displays the timestamp converted to the local timezone.

The number to the right a property value is the number of extracted entities that have that property value:

Facet values with the count 0 are not displayed.

When you enter values, the value is enclosed in quotes; the value inside the quotes must match the metadata exactly. For example:

- Typing "sample_*" in the originalName property returns only entities whose names match that exact string.
- To perform a wildcard search, type the wildcard string in the Search box. For example, typing the string "sample_*" in the Search box returns all entities with "sample_" at the beginning of their original name.

When you construct search strings with filters, use parentheses () to specify multiple values of a property. Add multiple properties by using the + operator. For example, entities of type HDFS or Hive that are of type file or directory:

```
+(sourceType:hdfs sourceType:hive) +(type:file type:directory)
```

and:

```
(...sourceType:hdfs +created:[NOW/DAY-30DAYS TO NOW/DAY+1DAY]) sourceType:hive
```

Grouping Search Results Using Metadata

Using the Group by option, you can organize the search results into groups based on the common properties of the entities in your search, including managed, custom, and some technical metadata. When you choose a group by property, the search panel lists the property values and shows how many search results correspond to that value.
From the Group by view, you can:

- Click the number for a group to show the search results for that group.

![Click the number for a group example](image1)

- Click the "x" on the Group by breadcrumb to remove the groupings.

![Click the x to remove groupings example](image2)

When you group results by a numeric or date property, you can also choose to "drill down" into a single group: the results from that group are grouped into smaller buckets. Hover over the group row to show the **Drill into group** control.

![Drill into group example](image3)
The Group by view shows at most 50 values of a property.

**Saving Searches**

1. Specify a search string or set of filters.
2. To the right of the Search box, select **Actions > Save, Actions > Save Search_name**, or **Actions > Save As**.
3. If you have not previously saved the search, specify a name and click **Save**.

**Reusing a Saved Search**

1. To the right of the Search box, select **Actions > View saved searches**. A label with the saved search name is added under the Search box.
2. Click the saved search name. The breadcrumbs and full query if displayed are updated to reflect the saved search, and the search results are refreshed immediately.

**Performing Actions on Entities**

**Required Role:** **Policy Editor** (or Full Administrator)

**Moving an HDFS Entity and Moving an HDFS Entity to Trash**

You can move an HDFS entity to another HDFS location, and to HDFS trash. To perform such actions, you must be a member of a user group that has the appropriate access to HDFS files.

**Note:** To move HDFS entities to another file system such as S3, consider designating an HDFS directory as an archive location, define a Navigator policy to move the HDFS entities to that directory, and define a CRON job or other process to regularly move the content of the archive directory to the non-HDFS location.

You can also schedule a move or move to trash in a policy.

After an HDFS entity is deleted, its record stays in Navigator until it is purged, which is 60 days by default. For information on what gets purged and when, see **Scheduling the Purge Process** on page 80.

1. Open your browser.
2. Navigate to the host within the cluster running the Cloudera Navigator Metadata Server role using the correct host name and port for your instance:
   
   `http://fqdn-1.example.com:7187/login.html`

   The login prompt displays.
3. Log in to the Cloudera Navigator console using the credentials assigned by your administrator.
4. Run a search in the Cloudera Navigator console.
5. Click an HDFS entity link returned in the search. The entity Details tab displays.
6. To the left of the Details tab, select **Actions > Move...** or **Actions > Move to Trash**.
7. For a move, specify the target path.
8. Click **Run Action**.

   When you delete a file, after a short delay the file displays a Deleted badge.

**Viewing Command Action Status**

1. Access the Cloudera Navigator console using the correct host name and port for your instance:

   `http://fqdn-1.example.com:7187/login.html`
The login prompt displays.

2. Log in to the Cloudera Navigator console using the credentials assigned by your administrator.

3. In the top right, click **username > Command Actions**. The Command Actions status page displays with a list of actions performed and the policy that caused the action, if applicable.

4. If an action failed, a View Log button displays, which you can click to view the error message associated with the failure.

**Viewing an Entity in Hue**

If the Hue component is running on the cluster view some entities using Hue and related tools, as follows:

<table>
<thead>
<tr>
<th>File Browser</th>
<th>HDFS directories and files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hive Metastore Manager (HMS)</td>
<td>Hive database and tables</td>
</tr>
<tr>
<td>Job Browser</td>
<td>YARN, Oozie</td>
</tr>
</tbody>
</table>

1. Access the Cloudera Navigator console using the correct host name and port for your instance:

   http://fqdn-1.example.com:7187/login.html

   The login prompt displays.

2. Log in to the Cloudera Navigator console using the credentials assigned by your administrator.

3. Run a search in the Cloudera Navigator console.

4. Do one of the following:

   - Search results
     1. Click the View in Hue link in a search result entry.
   - Entity details
     1. Click an entity link returned in the search. The entity Details tab displays.
     2. To the left of the Details tab, select Actions > View in Hue.

   The entity displays in the supported Hue application.

**Using Policies to Automate Metadata Tagging**

Cloudera Navigator lets you automate the application of metadata to specific entity classes using the policies you define. The policy specifies the actions to be performed by Navigator Metadata Server and the conditions under which to apply them. For data stewards who want to facilitate self-service discovery in their organizations, Cloudera Navigator’s metadata policy feature provides a robust mechanism for streamlining the metadata management tasks. For example, you can define policies that:

- Add managed metadata, tags, and custom metadata to entities as they are ingested by the cluster
- Move entities of specific class to a specific target path, or to the trash
- Send messages to a JMS message queue for notifications. This requires configuring the JMS server on the Cloudera Management Service. See [Configuring a JMS Server for Policy Messages](#) for details.

Messages sent to JMS queues are formatted as JSON and contain the metadata of the entity to which the policy should apply and the policy’s specified message text. For example:

```json
{
    "entity": "entity_property",
    "userMessage": "some message text"
}
```
Policies are executed in the home directory of their creator and can only take actions (run commands) for which the creator has privileges. At runtime, a policy fails if the user account of the policy creator does not have privileges to perform all the commands specified in the policy.

Policy ownership can be changed by cloning the policy:

- Log in to the Cloudera Navigator console using the account name to which you want to transfer ownership.
- Clone the policy.
- Log out from this account, and then log in again using account identity of original policy owner.
- Delete or disable the original policy.

**Note:** To ensure that command actions do not fail, policies containing command actions should be created by data stewards who are members of a user group that has the appropriate access to HDFS files.

Certain actions can be specified using Java expressions. See Metadata Policy Expressions on page 48 for details.

**Creating Policies**

**Required Role:** Policy Editor (or Full Administrator)

These steps begin from the Cloudera Navigator console.

1. Click the Policies tab.
2. Click the New Policy button.
3. Click the Enable checkbox.
4. Enter a meaningful name for the policy.
5. In the Search Query field, specify the search query that defines the class of entities to which the policy should apply.
6. Click the Test Query link to see a list of results based on the query and revise the query as needed.
7. Enter a Policy Description to document some of the functional aspects of this policy in the context of your organization. This field is optional but it is recommended for use especially if your organization has many policies defined for use by different teams, departments, and so on.
8. For a policy that includes Java expressions, enter the classes that the expressions will use in the Import Statements field. See Metadata Policy Expressions on page 48 for details, including examples and a class reference.
9. Choose the schedule for applying the policy:
   - **On Change** - When the entities matching the search string change.
   - **Immediate** - When the policy is created.
   - **Once** - At the time specified in the Start Time field.
   - **Recurring** - At recurring times specified by the Start and End Time fields at the interval specified in the Interval field.

   For Once and Recurring fields, specify dates and times using the calendar, time, and date tools available for the setting.

10. Follow the appropriate procedure for the actions performed by the policy:

   - **Metadata Assignments**: Specify the custom metadata or managed metadata to be assigned. Optionally, you can define Java expressions for fields in the policy that support them. Check the **Expression** checkbox to select this capability. The following fields support expressions:
     - Name
     - Description
     - Managed Metadata
     - Key-Value Pairs

   - **Command Actions**: Select Add Action > Move to Trash or Add Action > Move. For a move, specify the location to move the entity to in the Target Path field. If you specify multiple actions, they are run in the order in which they are specified.

   Command actions are supported only for HDFS entities. If you configure a command action for unsupported entities, a runtime error is logged when the policy runs.

   See Viewing Command Action Status on page 44.

   - **JMS Notifications**: If not already configured, configure a JMS server and queue. Specify the queue name and message. Optionally, check the **Expression** checkbox and specify a policy expression for the message.

11. Click **Save**.

Viewing Policies

**Required Role:** Policy Viewer (or Policy Editor, or Full Administrator)

1. Log in to the Cloudera Navigator console.
2. Click the Policies tab.
3. In a policy row, click a policy name link or select Actions > View. The policy detail page is displayed.

   You can also edit, copy, or delete a policy from the policy details page by clicking the Actions button.

<table>
<thead>
<tr>
<th>Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td>hdfsImmediatePolicy</td>
</tr>
</tbody>
</table>

**Status:** Enabled

**Search Query:** filesystemPath:’/tmp/policy_hdfs_data/testfile1’ AND sourceType:hdfs AND deleted: false

**Policy Description:**

**Last Run On:** Friday, December 9th 2016, 6:47 am

**Enabling and Disabling Policies**

As a policy administrator, you can manage access to policies by enabling and disabling them.
Required Role: **Policy Editor** (or Full Administrator)

1. Log in to the Cloudera Navigator console.
2. Click the Policies tab.
3. In a policy row, click a policy name link or select Actions > Enable or Actions > Disable.

**Copying and Editing a Policy**

If you have an existing policy that you want to use as a template for another similar property, you can copy it and then make any required adjustments. You can also edit existing policies if you need to make changes to it.

Required Role: **Policy Editor** (or Full Administrator)

1. Log in to the Cloudera Navigator console.
2. Click the Policies tab.
3. In a policy row, select Actions > Copy or Actions > Edit. You can also click the policy row and then on the policy details page, select Actions > Copy or Actions > Edit.
4. Edit the policy name, search query, or policy actions.
5. Click Save.

**Deleting Policies**

Required Role: **Policy Editor** (or Full Administrator)

1. Log in to the Cloudera Navigator console.
2. Click the Policies tab.
3. In a policy row, select Actions > Delete and OK to confirm.

**Metadata Policy Expressions**

A *metadata policy expression* is a Java expression that uses Java expressions instead of string literals to define properties for a metadata extraction policy. Properties that can be defined using Java expressions include entity names and descriptions, managed metadata, key-value pairs, and JMS notification messages. Declare classes for the expression to use in the policy’s import statement. The Java expressions defined to apply policies must evaluate to strings.

The Cloudera Navigator console displays an Expression check box near entities that can use Java expressions. Click the help icon for a pop-up example of an expression:

```java
Using Expressions
An expression example is below. Remember to import appropriate namespaces.

entity.get(FSEntityProperties.ORIGINAL_NAME, Object.class) + " " + entity.get(FSEntityProperties.TYPE, Object.class)
```

Expressions can support the following:

- **Metadata Assignments**
  - Name
  - Description
  - Managed Metadata
  - Key-Value Pairs
- **JMS Notification Messages**
Including Entity Properties in Policy Expressions

To include entity properties in property expressions, use the `entity.get` method and pass the property and return type as follows:

```java
entity.get(EnumTypeProperties.PROPERTY, return_type)
```

The `EnumTypeProperties.PROPERTY` and respective values for `return_type` are detailed in full in the Entity Property Enum Reference below. The `EnumType` class names are:

- `FSEntity`
- `HiveColumn`
- `HiveDatabase`
- `HivePartition`
- `HiveQueryExecution`
- `HiveQueryPart`
- `HiveQuery`
- `HiveTable`
- `HiveView`
- `JobExecution`
- `Job`
- `WorkflowInstance`
- `Workflow`
- `PigField`
- `PigOperationExecution`
- `PigOperation`
- `PigRelation`
- `SqoopExportSubOperation`
- `SqoopImportSubOperation`
- `SqoopOperationExecution`
- `SqoopQueryOperation`
- `SqoopTableExportOperation`
- `SqoopTableImportOperation`

For any operation based on return type (`return_type`), use the return type listed in Entity Property Enum Reference on page 50. Each `PROPERTY` listed for a given `EnumType` has its data type (`return_type`) specified following the Java comment (`//`).

For example, the return type for the `ORIGINAL_NAME` property of the `FSEntityProperties` class is `java.lang.String`, which means you can use `String` methods such as `toLowerCase()` to modify the returned value:

```java
entity.get(FSEntityProperties.ORIGINAL_NAME, String.class).toLowerCase()
```

**Note:** For expressions that do not require specific type, use `Object.class` as the return type.
Metadata Policy Expression Examples

Here are some short examples showing how to use Java expressions for metadata policies. The examples include the import statements to enter into the Import Statements field when creating policies in the Cloudera Navigator console.

- Set a filesystem entity name to the original name concatenated with the entity type:

```java
entity.get(FSEntityProperties.ORIGINAL_NAME, Object.class) + " " +
entity.get(FSEntityProperties.TYPE, Object.class)
```

Import Statements:
```
import com.cloudera.nav.hdfs.model.FSEntityProperties;
```

- Add the entity's creation date to the entity name:

```java
entity.get(FSEntityProperties.ORIGINAL_NAME, Object.class) + " - " +
new SimpleDateFormat("yyyy-MM-dd").format(entity.get(FSEntityProperties.CREATED,
Instant.class).toDate())
```

Import Statements:
```
import com.cloudera.nav.hdfs.model.FSEntityProperties; import java.text.SimpleDateFormat; import org.joda.time.Instant;
```

- Set the key-value pair: retain_util-seven years from today's local time:

```java
new DateTime().plusYears(7).toLocalDateTime().toString("MMM dd yyyy", Locale.US)
```

Import statements:
```
import org.joda.time.DateTime; import java.util.Locale;
```

Entity Property Enum Reference

The Java enumerations for retrieving properties from each entity type are listed below.

```java
com.cloudera.nav.hdfs.model.FSEntityProperties
public enum FSEntityProperties implements PropertyEnum {
    PERMISSIONS, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    SIZE, // Return type: java.lang.Long
    OWNER, // Return type: java.lang.String
    LAST_MODIFIED, // Return type: org.joda.time.Instant
    SOURCE_TYPE, // Return type: java.lang.String
    DELETED, // Return type: java.lang.Boolean
    FILE_SYSTEM_PATH, // Return type: java.lang.String
    CREATED, // Return type: org.joda.time.Instant
    LAST_ACCESSED, // Return type: org.joda.time.Instant
    GROUP, // Return type: java.lang.String
    MIME_TYPE, // Return type: java.lang.String
    DELETE_TIME, // Return type: java.lang.Long
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}
```

```java
com.cloudera.nav.hive.model.HiveColumnProperties
public enum HiveColumnProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
```
com.cloudera.nav.hive.model.HiveDatabaseProperties
public enum HiveDatabaseProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
    ORIGINAL_DESCRIPTION, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    DELETED, // Return type: java.lang.Boolean
    FILE_SYSTEM_PATH, // Return type: java.lang.String
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.hive.model.HivePartitionProperties
public enum HivePartitionProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
    ORIGINAL_DESCRIPTION, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    DELETED, // Return type: java.lang.Boolean
    FILE_SYSTEM_PATH, // Return type: java.lang.String
    CREATED, // Return type: org.joda.time.Instant
    LAST_ACCESS, // Return type: org.joda.time.Instant
    COL_VALUES, // Return type: java.util.List
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.hive.model.HiveQueryExecutionProperties
public enum HiveQueryExecutionProperties implements PropertyEnum {
    SOURCE_TYPE, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    ENDED, // Return type: org.joda.time.Instant
    INPUTS, // Return type: java.util.Collection
    OUTPUTS, // Return type: java.util.Collection
    STARTED, // Return type: org.joda.time.Instant
    PRINCIPAL, // Return type: java.lang.String
    WF_INST_ID, // Return type: java.lang.String
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.hive.model.HiveQueryPartProperties
public enum HiveQueryPartProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
Cloudera Navigator Metadata

```java
public enum HiveQueryProperties implements PropertyEnum {
    SOURCE_TYPE, // Return type: java.lang.String
    INPUTS, // Return type: java.util.Collection
    OUTPUTS, // Return type: java.util.Collection
    QUERY_TEXT, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

public enum HiveTableProperties implements PropertyEnum {
    OWNER, // Return type: java.lang.String
    INPUT_FORMAT, // Return type: java.lang.String
    OUTPUT_FORMAT, // Return type: java.lang.String
    DELETED, // Return type: java.lang.Boolean
    FILE_SYSTEM_PATH, // Return type: java.lang.String
    COMpressed, // Return type: java.lang.Boolean
    PARTITION_COL_NAMES, // Return type: java.util.List
    CLUSTERED_BY_COL_NAMES, // Return type: java.util.List
    SORT_BY_COL_NAMES, // Return type: java.util.List
    SER_DE_NAME, // Return type: java.lang.String
    SER_DE_LIB_NAME, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    CREATED, // Return type: org.joda.time.Instant
    LAST_ACCESSED, // Return type: org.joda.time.Instant
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

public enum HiveViewProperties implements PropertyEnum {
    DELETED, // Return type: java.lang.Boolean
    QUERY_TEXT, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    CREATED, // Return type: org.joda.time.Instant
    LAST_ACCESSED, // Return type: org.joda.time.Instant
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

public enum JobExecutionProperties implements PropertyEnum {
    SOURCE_TYPE, // Return type: java.lang.String
    JOB_ID, // Return type: java.lang.String
    ENDED, // Return type: org.joda.time.Instant
    INPUT_RECURSIVE, // Return type: boolean
    TYPE, // Return type: java.lang.String
}
```
com.cloudera.nav.mapreduce.model.JobProperties
public enum JobProperties implements PropertyEnum {
    ORIGINAL_NAME, // Return type: java.lang.String
    INPUT_FORMAT, // Return type: java.lang.String
    OUTPUT_FORMAT, // Return type: java.lang.String
    OUTPUT_KEY, // Return type: java.lang.String
    OUTPUT_VALUE, // Return type: java.lang.String
    MAPPER, // Return type: java.lang.String
    REDUCER, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.oozie.model.WorkflowInstanceProperties
public enum WorkflowInstanceProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    CREATED, // Return type: org.joda.time.Instant
    JOB_ID, // Return type: java.lang.String
    STATUS, // Return type: java.lang.String
    ENDED, // Return type: org.joda.time.Instant
    INPUTS, // Return type: java.util.Collection
    OUTPUTS, // Return type: java.util.Collection
    STARTED, // Return type: org.joda.time.Instant
    PRINCIPAL, // Return type: java.lang.String
    WF_INST_ID, // Return type: java.lang.String
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.oozie.model.WorkflowProperties
public enum WorkflowProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.pig.model.PigFieldProperties
public enum PigFieldProperties implements PropertyEnum {
    TYPE, // Return type: java.lang.String
com.cloudera.nav.sqoop.model.SqoopImportSubOperationProperties
public enum SqoopImportSubOperationProperties implements PropertyEnum {
    DB_COLUMN_EXPRESSION, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    INPUTS, // Return type: java.util.Collection
    FIELD_INDEX, // Return type: int
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.sqoop.model.SqoopOperationExecutionProperties
public enum SqoopOperationExecutionProperties implements PropertyEnum {
    source_type, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    ENDED, // Return type: org.joda.time.Instant
    INPUTS, // Return type: java.util.Collection
    OUTPUTS, // Return type: java.util.Collection
    STARTED, // Return type: org.joda.time.Instant
    WF_INST_ID, // Return type: java.lang.String
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.sqoop.model.SqoopQueryOperationProperties
public enum SqoopQueryOperationProperties implements PropertyEnum {
    QUERY_TEXT, // Return type: java.lang.String
    INPUTS, // Return type: java.util.Collection
    DB_USER, // Return type: java.lang.String
    DB_URL, // Return type: java.lang.String
    OPERATION_TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}

com.cloudera.nav.sqoop.model.SqoopTableExportOperationProperties
public enum SqoopTableExportOperationProperties implements PropertyEnum {
    DB_TABLE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    DB_USER, // Return type: java.lang.String
    DB_URL, // Return type: java.lang.String
    OPERATION_TYPE, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}
com.cloudera.nav.sqoop.model.SqoopTableImportOperationProperties
public enum SqoopTableImportOperationProperties implements PropertyEnum {

    DB_TABLE, // Return type: java.lang.String
    DB_WHERE, // Return type: java.lang.String
    SOURCE_TYPE, // Return type: java.lang.String
    DB_USER, // Return type: java.lang.String
    DB_URL, // Return type: java.lang.String
    OPERATION_TYPE, // Return type: java.lang.String
    TYPE, // Return type: java.lang.String
    WF_IDS, // Return type: java.util.Collection
    NAME, // Return type: java.lang.String
    ORIGINAL_NAME, // Return type: java.lang.String
    USER_ENTITY, // Return type: boolean
    SOURCE_ID, // Return type: java.lang.String
    EXTRACTOR_RUN_ID, // Return type: java.lang.String
    PARENT_PATH; // Return type: java.lang.String
}
Lineage Diagrams

When it comes to making important business decisions, filing financial records, or complying with all manner of regulations, organizations need verifiable data. Public corporations in the United States must be able to prove to the IRS or to the SEC that values supplied in balance sheets and income statements are legitimate, for example. In pharmaceuticals research, data collected over multi-year clinical trials that may be combined with anonymous patient statistics must be able to be traced to data sources.

Cloudera Navigator lineage diagrams are designed to enable this type of verification. Lineage diagrams can reveal the provenance of data entities—the history of the data entity, back to its source with all changes along the way to its present form.

A lineage diagram is a directed graph that shows how data entities are related and how an entity may have changed over its lifetime in the cluster. Cloudera Navigator uses the metadata associated with entities contained in the cluster to render lineage diagrams that can trace data sources back to the column level and show relations and the results of transformations. See Lineage Diagram icons on page 159 for details about the line colors, styles, and icons used to render lineage diagrams, some of which are explained in the context of the examples below.

Using the Lineage View

For information about how lineage relationships are formed, see Generating Lineage Diagrams on page 67.

Exploring Lineage Diagrams

Required Role: Metadata & Lineage Viewer (or Managed & Custom Metadata Editor, or Full Administrator)

Lineage diagrams are accessible in various ways in the Cloudera Navigator console. This example uses Search to first find a specific entity and then display its lineage diagram. Use this approach if you know the name, data type, source type, or another property value for a specific entity. The example uses the Hive sample data warehouse (installed by default) and assumes the following query has been run:

```sql
SELECT sample_07.description, sample_07.salary FROM sample_07
WHERE (sample_07.salary > 100000)
ORDER BY sample_07.salary DESC LIMIT 1000
```

The query returns values from description and salary columns for salaries over 100,000 from the sample_07 table. Because Cloudera Navigator automatically collects metadata from entities, it can render lineage diagrams on any data contained in the cluster, including operations, such as the SELECT query of the example.

To display the lineage diagram based on this query in Cloudera Navigator console:

- Click the Search tab.
- Click Hive and Operation filters limit the scope of the search.
- Type the word salary or sample_07 to find the query created above.
- Find the returned entity that contains the query and click on it to display its Details page.
- Click the query and then click the Lineage tab to display the lineage diagram.
The columns identified in the `select` clause use solid directed lines to the source table (shown with the Hive Table icon) and a dashed line to the source column of the `where` clause (shown with the Operation icon).

Expanding Entities

The example shows how one lineage diagram can expand to another. The first image shows a YARN operation (PigLatin:DefaultJobName) that runs an associated Pig script of the same name (DefaultJobName). The solid lines represent data flow relationships, from the source file (in the ord_us_gcb_crd_crs-fdr-sears folder) through the script execution, to the destination (in folder tmp137071676). The YARN operation PigLatin:DefaultJobName is rendered using a dashed (rather than solid) line because it is the starting point for this particular lineage diagram. In lineage diagrams, lines change to the color blue when they are selected. Containment is signalled using the plus icon:

Click the plus icon to expand a parent entity and display child entities. For example, expanding the Pig script reveals the two Pig tables (ACT_FILE, ACT_FILE_COL_CNT_ERR_ROWS) referenced by the script. The solid directional line between the two tables indicates the data flow relationship between these two tables.

Adjusting the Lineage Layout for Readability

Lineage diagrams displayed in the Cloudera Navigator console can be manipulated by as follows:

- Click and drag entities outside their parent boxes.
- Use the plus-minus control displayed in the lineage diagram or the mouse scroll wheel to expand and shrink (zoom in, zoom out) the image size
- Relocate the lineage diagram within the pane using click-hold-drag gesture.
Filtering Lineage Diagrams

Lineage diagrams can render faster when filters are applied. Filters can limit or specify the entities displayed.

<table>
<thead>
<tr>
<th>Filter</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>The <strong>Lineage Options</strong> default selection applies the Latest Partition and Operation filter, which displays only the most recent partitions and operations. For example, if Hive partitions are created daily, the filter displays only the latest partition.</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** This filter applies only to metadata collected with Navigator Metadata Server installed with Cloudera Manager version 5.12 or later. Operations against multiple partitions that were collected by earlier versions will not collapse into a single partition.

Lineage diagram of the `sample_09` table filtering deleted items only.

**Control Flow Relations.** The operation is collapsed and control flow links are hidden.
<table>
<thead>
<tr>
<th>Filter</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>The <strong>Only Upstream</strong> filter displays only input (upstream) entities and links. The <strong>Only Downstream</strong> filter displays only output (downstream) entities and links. The operation is collapsed and only upstream (or downstream) entities and links display. The output table is hidden.</td>
<td></td>
</tr>
</tbody>
</table>

Here, the operation is collapsed and only downstream entities and links are shown. The input tables are hidden.

The **Operations** filter collapses operations into green links between entities.

The **Deleted Entities** filter displays entities that have been deleted but maintains relations to other entities.
Exploring Hidden Entities in a Lineage Diagram

Lineage diagrams use the **hidden icon** to indicate areas in the diagram available for further exploration. The basic mouse gestures to use to traverse a lineage diagram with hidden information are as follows (there are many alternative approaches, including simply double-clicking the hidden icon):

1. Hover over the hidden icon (note the hand floating at the top of the hidden icon) to display an information text (box, upper-right) with the link that provides access to lineage for the additional entities.

2. Click the hidden icon to select it.

3. Click the **view the lineage** link to display a new lineage diagram contained at that intersect point. The green line is a summary line that contains

![Diagram showing hidden entities and linkage](image)

This shows that the sample_08 table is actually contained in a folder, sample_08.

Finding Specific Entities in Lineage Diagrams

Cloudera Navigator's Search function can be used in the context of an open lineage diagram. This is useful especially when lineage diagrams have hidden entities that are may not be visible.

1. In the **Search** box at the right of the diagram, type an entity name. A list of matching entities displays below the box.

2. Click an entity in the list. A blue box is drawn around the entity and its details display in a box below the Search box.

![Diagram showing search results](image)
3. Click the Show link next to the entity. The selected entity moves to the center of the diagram.

4. To view the lineage of the selected entity, click the View Lineage link in the entity details box.

Displaying a Template Lineage Diagram

A template lineage diagram contains template entities, such as jobs and queries, that can be instantiated, and the input and output entities to which they are related.

To display a template lineage diagram:

1. Perform a metadata search.
2. In the list of results, click an entity. The entity Details page displays. For example, when you click the sample_09 result entry:

![Entity Details Page]

the Search screen is replaced with a Details page that displays the entity property sheet:

![Entity Property Sheet]

3. Click the Lineage tab. For example, clicking the Lineage tab for the sample_09 table displays the following lineage diagram:

![Lineage Diagram]

This example shows the relations between a Hive query execution entity and its source and destination tables:

![Hive Query Lineage]

Click the plus icon to display the columns and lines connecting the source and destination columns display:
Displaying an Instance Lineage Diagram

An instance lineage diagram displays instance entities, such as job and query executions, and the input and output entities to which they are related. To display an instance lineage diagram:

1. Perform a search and click a link of type Operation.
2. Click a link in the Instances box.
3. Click the Lineage tab.

Displaying the Template Lineage Diagram for an Instance Lineage Diagram

To browse from an instance diagram to its template:

1. Display an instance lineage diagram.
2. Click the Details tab.
3. Click the value of the Template property to go to the instance’s template.

Using Lineage to Display Table Schema

Required Role: Metadata & Lineage Viewer (or Managed & Custom Metadata Editor, or Full Administrator)

A table schema contains information about the names and types of the columns of a table.

A Kite dataset ingested into HDFS contains information about the names and types of the fields in an HDFS Avro or Parquet file used to create the dataset.

Displaying Hive, Impala, and Sqoop Table Schema

1. Use the Cloudera Navigator console to search for entities of source type Hive and type Table.
2. In the list of results, click a result entry. The table schema displays in the Details tab.

Displaying Pig Table Schema

1. Use the Cloudera Navigator console to search for entities of source type Pig.
2. In the list of results, click a result entry of type Table. The table schema displays in the Details tab.
Displaying HDFS Dataset Schema

If you ingest a Kit data into HDFS, you can view the schema of the dataset. The schema is represented as an entity of type Dataset and is implemented as an HDFS directory.

For Avro datasets, primitive types such as null, string, int, and so on, are not separate entities. For example, if you have a record type with a field A that's a record type and a field B that's a string, the subfields of A become entities themselves, but B has no children. Another example would be if you had a union of null, string, map, array, and record types; the union has 3 children - the map, array, and record subtypes.

To display an HDFS dataset schema:

1. Use the Cloudera Navigator console to search for specific Dataset.
2. Click a result entry. The dataset schema displays in the Details tab.

Stocks Schema

1. Use the Stocks Avro schema file:

```json
{
  "type": "record",
  "name": "Stocks",
  "namespace": "com.example.stocks",
  "doc": "Schema generated by Kite",
  "fields": [
    {
      "name": "Symbol",
      "type": ["null", "string"],
      "doc": "Type inferred from 'AAIT'"
    },
    {
      "name": "Date",
      "type": ["null", "string"],
      "doc": "Type inferred from '28-Oct-2014'"
    },
    {
      "name": "Open",
      "type": ["null", "double"],
      "doc": "Type inferred from '33.1'"
    },
    {
      "name": "High",
      "type": ["null", "double"],
      "doc": "Type inferred from '33.13'"
    },
    {
      "name": "Low",
      "type": ["null", "double"],
      "doc": "Type inferred from '33.1'"
    },
    {
      "name": "Close",
      "type": ["null", "double"],
      "doc": "Type inferred from '33.13'"
    },
    {
      "name": "Volume",
      "type": ["null", "long"],
      "doc": "Type inferred from '400'"
  ]
}
```

and the `kite-dataset` command to create a Stocks dataset:

```bash
kite-dataset create dataset:hdfs:/user/hdfs/Stocks -s Stocks.avsc
```

The following directory is created in HDFS:
2. In search results, the Stocks dataset appears as follows:

3. Click the **Stocks** link. The schema displays at the right of the Details tab.

Each subfield of the Stocks record is an entity of type Field.

4. Then use the **kite-dataset** command to import structured data:

   ```
   kite-dataset csv-import ./Stocks.csv dataset:hdfs:/user/hdfs/Stocks --no-header
   ```

   where **Stocks.csv** is:

   ```
   AAPL,20150206,120.02,120.25,118.45,118.93,43372000
   AAPL,20150205,120.02,120.23,119.25,119.94,42246200
   GOOG,20150304,571.87,577.11,568.01,573.37,1713800
   GOOG,20150303,570.45,575.39,566.52,573.64,1694100
   GOOG,20150302,560.53,572.15,558.75,571.34,2118400
   GOOG,20150209,528,532.526.02,527.83,1264300
   GOOG,20150208,527.64,537.2,526.41,531,1744600
   GOOG,20150205,523.79,528.5,522.09,527.58,1844700
   FB,20150304,79.3,81.15,78.85,80.9,28014500
   ```
Generating Lineage Diagrams

Navigator Metadata Server uses the metadata it collects to generate lineage graphs. These graphs tell you what data and processes were used to generate a given data asset.

This documentation describes

- Where the lineage information comes from
- How the lineage varies based on the source of the metadata
- Life cycle of lineage info as assets change on the cluster

Lineage Generation Architecture

Navigator Metadata Server collects metadata from CDH services, creates records in Solr for each of the data assets and operations, then uses the operation metadata to build relations among the data assets. You see the relations among data assets in the Navigator Console as a lineage graph. Here’s the process in a little more detail:
1. Extractors collect metadata from services in the cluster.

The extractors are components within Navigator that communicate with each service that Navigator supports. For example, for Navigator Metadata Server to collect HDFS metadata, the Navigator HDFS extractor talks to the namenode to request file system information.

There are two types of extractors:
- Pull extractors (CDX, HDFS, YARN, Oozie, HMS). The extractor requests metadata from the service about every 10 minutes. The CDX extractor pulls metadata from S3.
- Push extractors (Spark, Impala, Hive Server 2, Sqoop). The service saves metadata to the local file system for every operation; the Cloudera Manager agent running on the same host passes the metadata to Navigator Metadata Server about every five seconds.

2. Navigator indexes each data asset and operation.

For each new data asset and operation it finds in the extracted metadata, Navigator Metadata Server creates an entry in its Solr index. These entries appear in `nav_elements` core. The most recently extracted metadata is always used to update any existing entity entry. In addition, Navigator Metadata Server creates a `nav_relation` entry in the Solr index to store relationship information, such as between a directory and a file or subdirectory or between a table and the directory backing it. Navigator saves a variety of relationship information:
- Logical - physical: Hive table and the HDFS directory that contains the data.
- Instance of: Operation and operation execution.
- Data flow: Query operation that refers to input or output tables.
- Control flow: Column level details in an operation that refers to specific column or field details in a table, view, or file.

If Navigator doesn’t have all the information it needs to fill in both ends of a relationship, it creates an "unlinked" relation. For example, when an HMS table is extracted, but the corresponding HDFS directory is not already in the index, Navigator will create an unlinked relation for the table that will be filled in later with the directory metadata.

3. Linker fills in missing relation endpoints.
After it receives metadata from an extraction, Navigator Metadata Server runs a process ("linker") that uses the new metadata to fill in unlinked relations.

4. Navigator console displays the relations as a lineage diagram.

The Navigator console reads `nav_relation` to show entity relationships in a directed graph, where the graph flows from input to output, left to right. You can choose how much detail you want to see in the graph, including showing the operations that give details on how data was transformed.

The entities shown in a lineage picture are determined by the service where the metadata came from. The lineage for data assets includes other data assets and the operations that relate them; lineage for query and compute operations include existing data assets used as inputs or outputs.

For more about how to use the lineage view to show more information, see Exploring Lineage Diagrams on page 57.

**Lineage Life-cycle**

The contents of a lineage diagram are determined by what metadata is extracted from services. If an object doesn't exist when the extraction occurs, it will not be reflected in the lineage diagram.

**Metadata extraction operation and timing**

The appearance of entities in a lineage graph can depend on metadata from multiple sources, each of which is extracted independently. To understand lineage generation, it is important to understand when extractors run and any dependencies on their success.

Navigator triggers metadata extraction for each type of pull extractor every 10 minutes after the completion of the previous extraction. The extractor operations run until the extractor processes all the metadata provided by the service; an extraction could be running for minutes or hours depending on the volume of metadata being extracted and the resources available. HDFS extraction—generally the largest volume of metadata—performs an incremental extraction, so after it runs once, subsequent extractions are significantly faster.

Push extractors work differently: instead of waiting for Navigator to ask for the metadata, the services write out metadata when an event occurs; the Cloudera Manager agent on the host polls every 5 seconds and forwards whatever metadata it finds to Navigator Metadata Server.

The extraction processes are limited by the amount of memory available for Solr to generate the index from the incoming metadata and the Navigator Metadata Server heap. If the extractor fails to connect to its service, it will try again at the next interval.

Navigator runs the linking process after each extractor completes processing an extraction. The linking process can also take minutes or hours, depending on the volume of data being processed and resources available.

**Deleted data assets**

Each metadata extraction from a service is a snapshot of the objects managed by that service at a given time. If one extraction includes a reference to a specific table and the following extraction does not, Navigator Metadata Server
flags the entity for that table as deleted. These relations are hidden in lineage diagrams. You can show them again by enabling the Deleted Entities in the lineage view options. The deleted entities are indicated with a line through their name.

**Metadata Purge and Lineage**

Metadata purge reviews lineage relations where at least one endpoint of the relation is a candidate for removal. A data asset entity would be flagged as deleted; operation executions would have to have been executed earlier than the purge threshold date. What gets purged is slightly different among the different types of objects. See [What Metadata is Purged?](#) on page 81 for purge details. Here’s how purge affects lineage:

- "Instance of" relations between operation and operation executions are removed when the operation execution was executed earlier than the threshold date for purge.
- Data flow relations for operations are removed when:
  - The operation doesn't have an output entity; that is, the operation does not produce a table or view or other entities produced by Sqoop jobs or associated with Pig tables.
  - The associated operation executions are older than the purge threshold date.
- Parent child relations between HDFS files and their parent directory are only removed when the directory has been flagged as deleted longer than the purge threshold date.

The following items are not affected by purge:

- Hive tables and the operations that produce them are never removed, even if the tables were deleted before the purge threshold date.
- Logical-physical relations between Hive tables and HDFS directories are not removed, even if the tables were deleted before the purge threshold date.

**Temporary data assets**

Sometimes operations include data assets that are created and then deleted as part of the operation (or as part of a series of operations that occur close together in time). These temporary objects may or may not be captured as part of a Navigator pull extraction. The operation that produced the object is likely be extracted through a push extractor. When temporary objects are not captured, Navigator Metadata Server produces an unlinked relation based on the operation metadata. The technical metadata for the operation, such as query text, includes a reference to the temporary object; the object itself will not be listed in the Navigator console and will not appear in lineage diagrams.

For example, consider a Hive pipeline that writes data to a table, transforms the data and writes it to a second table, then removes the first table. The lineage content is different depending on whether or not metadata extraction captured the temporary table before it was dropped.

**If the Navigator-HMS extractor runs when the first table was present**, Navigator will create a metadata entity for the first table. When Navigator links the details of the operation with known entities, it will translate the operation details into a lineage diagram including the first and second tables. After the table is deleted and when the next extraction runs, Navigator Metadata Server flags the entity for the first table as deleted. The lineage diagram for the second table shows the first table as deleted (you may have to enable Deleted Entities in the lineage view options). The operation and operation executions still show in the lineage diagram.

**If the first table is created and deleted between Navigator-HMS extractions**, no metadata entity is created for the table, and the operation details don’t link up to an existing entity as an operation input (unlinked relation). If you look at the lineage diagram for the second table and turn on the option to show operations, you’ll see the operation that created the second table, but you won’t see the first table. In the lineage view, you can see the query run by the operation. That query text shows the first table in the FROM clause.
Example of lineage across temporary tables

This example shows how the lineage diagram changes depending on whether a temporary table was included in a metadata extraction. The following Impala queries were run in Hue to create temporary tables (`temp_sample_07` and `temp_sample_08`), then use those tables to load data into a third table (`sample_merge`):

```
create table tmp_sample_07 as select * from sample_07;
create table merge_sample as select * from tmp_sample_07;
create table tmp_sample_08 as select * from sample_08;
insert into merge_sample select * from tmp_sample_08;
drop table tmp_sample_08;
/* delay until after Navigator metadata extraction occurs */
drop table tmp_sample_07;
```

Assume that the `temp_sample_08` table was created and dropped between two metadata extractions and that the `temp_sample_07` DROP TABLE command is run only after metadata extraction occurred.

After extractions are run and linking occurs, the lineage picture for the `sample_merge` table shows an "upstream" relationship with `sample_07`.

After turning on the option for **Deleted Entities**, the temporary table `temp_sample_07` appears. It exists in the system because it was captured by the first metadata extraction and then it was deleted by the second extraction.

Turning on the option for **Operations** shows the queries that created and inserted rows into the `sample_merge` table. Note in particular that the operation shows for inserting data into the `sample_merge` table from the temporary table `temp_sample_08` even though the table was not captured by a metadata extraction.
If you select the insert operation, you can see the query details to identify the operation input.

If you wanted to further trace the origin of the temporary table, you can use the table name to search for operations that affected the temporary table by searching on the query text:

queryText:*temp_sample_08*

**Lineage Behavior by Service**

Lineage graphs follow some rules based on the kinds of metadata provided by services. This section describes how the metadata is formed into relations; the conventions for how the relations are displayed on the lineage diagram as listed in [Lineage Diagram Icons](#) on page 159. For a list of the specific operations that produce entities in Navigator, see [Service Metadata Entity Types](#) on page 173.

**Hive, Impala**

Lineage is generated for (most) operations against the Hive Metastore that create data assets or modify data, including operations from HiveServer2, Impala, and Impala queries on Kudu tables. Lineage is not produced for operations performed using the Hive Server 1 CLI. For a list of the specific operations captured by Navigator extractors, see [Hive Operations and Cloudera Navigator Support Matrix](#) on page 173.

For each query operation, a relation is created for each data asset referenced as an input or output (or both) in the query. The lineage diagrams include tables, views, and operations. The metadata indicates whether the data asset is an input or output.
The query information provided by Hive Metastore (HMS) includes query information that allows Navigator to show data assets referenced in joins or lookups. You can see these tables when you turn on Control Flow Relations.

When Navigator has column level information, the lineage diagram shows a + next to the name of the object. Opening the column level information shows you how columns in source tables contribute to columns in the destination tables. Columns referenced in joins or lookups are indicated with dashed lineage relations and appear when the view includes control flow information. (Move the cursor over a column to highlight only that column's lineage through all the objects.) Because the information is generated by HMS, column-level tracing is available regardless of the underlying compute engine.
Only database-layer objects appear in the lineage for query services: Navigator doesn’t show the logical-to-physical relationships such as the backing HDFS directory for each Hive table. If you want to know the location of the underlying HDFS files for a table, you can see this information in the Path property of the technical metadata for the table. Note that lineage shown for an HDFS file or directory, you will see the logical-to-physical relationship to Hive tables or views.

Only operations that include output appear in lineage for tables and views. To see a lineage diagram for operations such as SELECT that do not produce a data asset, view lineage for the operation. To see all such activity run against a specific table, you may find it useful to go the Audits tab in the Navigator console and filter on the table name.

Lineage for query services does not expose the compute engine used to perform the query; Hive operations appear the same whether they are run on MapReduce or Spark. This information is available in the technical metadata for the operation execution. Note that queries run directly in Spark are different because Spark provides different metadata than HMS.

**HDFS, S3**

Navigator collects metadata for file system entities. It does not collect metadata for file-system level operations, such as moves or copies. Lineage metadata appears when applications perform operations against file-system entities. For example, when a Spark job copies an HDFS directory, the lineage diagrams show the relationship between the source directory and the copied directory as connected by the Spark operation.

When HDFS directories are used as backing data for tables managed in Hive Metastore (including Impala tables), the lineage for the directory and any of the files included in the directory will show a logical-physical relation to the table.

**Figure 4: Logical Physical Relationship between HDFS File and Hive Table**

**MapReduce, Spark, Sqoop**

Navigator shows lineage diagrams for operations performed by compute engines such as MapReduce v1, YARN (MRV2), or Spark (all Spark 1 versions and Spark 2 starting with version 2.3).

Navigator shows lineage diagrams for operations performed by compute engines such as MapReduce v1, YARN (MRV2), or Spark (all Spark 1 versions). Lineage reflects data processed using Spark SQL, including the Datasets/Dataframe APIs. Other services such as Apache Sqoop contribute metadata that Navigator can use to create lineage diagrams as well.
For compute operations, a relation is created for each data asset referenced as an input or output (or both) in the operation. As in this image, the input and output relations are shown with arrows.

**Figure 5: Spark Lineage Shows All Data Assets Accessed by the Operation**

Data assets that are used for both output from and input to stages of the operation appear with two lineage lines, one with an arrow pointing into the entity; another pointing out. In the Navigator lineage diagram, entities that are both input and output to the same operation may be shown parallel to the operation rather than on the left or right.

The Navigator extractors for MapReduce and Spark services trace data lineage at the application level only. The result is that when you show the lineage diagram for a single column in a table that is affected by a Spark operation, the diagram shows other data assets that contribute to the column, but how the specific columns inside those assets contribute to the selected column is not shown.

Navigator shows all the lineage information available for Spark operations, but does not include SQL statements executed in Spark. This limitation is because Spark does not keep the SQL statement: the Spark parser creates the plan from the SQL statement and, from that point on, the SQL statement provided by the client is not retained as part of the metadata associated with the Spark job. For many Spark operations, SQL isn't part of the original request. For example: spark.read.parquet("/input").write.parquet("/output") generates lineage, but there is no SQL statement used at all.
Figure 6: Column Filtering for Spark Lineage Identifies Contributing Assets

Note: Lineage diagrams differ slightly between Spark 1 and Spark 2 operations. In Spark 1, table creation commands are not captured as a lineage relation: when a table is created and then used by another command, the table appears only as an input to the operation. In Spark 2, table creation commands are captured: the table shows as an output (creation) and an input to the operation.

YARN, Oozie

Lineage for resource management operations, such as YARN and Oozie, include lineage relations to other operations; the metadata for YARN and Oozie entities does not reveal specific data assets touched by the processing.

Navigator shows lineage diagrams for operations performed by resource management tools such as Oozie and YARN. Relations are created for each operation controlled by job managers. These lineage diagrams show processing stages, including the connection between a YARN operation and the operations triggered from YARN. Navigator captures the relation between a YARN job and the operations it triggers as "Logical Physical" relations: the physical operation is the YARN process; the operation it triggers, such as a Hive query or a MapReduce job, is the logical operation.
Viewing the lineage from the logical operation (in this case, a Hive query), the Logical Physical relations to the triggering operations are not shown.

Troubleshooting Lineage

When you don’t see the lineage relations you expect, look for the following possible issues:

Give time for extractors to run

Pull extractors run every ten minutes assuming the previous run completed. For large data sets, the extractor process may require more time to run. In addition, if there are connectivity issues or if the corresponding service is busy or not available, the extraction time can lengthen. For HMS in particular, the actual time required to extract all the metadata required to draw new lineage relations can be as much as an hour if the multiple extractors involved run offset from each other.

View lineage from the appropriate entity

If you want to see a logical-physical relation between an HDFS directory and a Hive table, you need to access lineage from the HDFS directory.

If you want to see all the times a query is run against a Hive table you may not see it in the Hive table lineage; if the operation doesn’t produce output, access the lineage from the operation, not the Hive table.

Are extractors running successfully?

Navigator won’t show lineage information if it’s not receiving it. Check to make sure that the extractor is running properly:

- Check the Navigator Metadata Server log for errors that would indicate that an extractor isn’t running. You can access Navigator Metadata Server logs from Cloudera Manager or from the command line on the host where the service is running. See Accessing Navigator Data Management Logs on page 84. Extractor status messages include the word "extractor", such as com.cloudera.nav.hive.extractor.HiveOperationExtractor.
- Check Cloudera Manager agent log to find out more detail about an error.

If you find an error in the Navigator Metadata Server log that indicates a problem with one of the push extractors, go to the Cloudera Manager agent log for the host indicated by the log error.
The Cloudera Navigator console provides an Administration menu for several different tasks, including setting up a schedule to remove stale and deleted metadata from the system ("purge"), setting up users and groups, and defining managed metadata.

For information on defining managed metadata, see Defining Properties for Managed Metadata on page 31.

Administering Navigator User Roles

Cloudera Navigator user roles can be assigned to groups that are managed by an external LDAP-compliant identity/authentication system (Active Directory, OpenLDAP) using the Role Management tab of the Cloudera Navigator console. The Role Management feature only becomes available in the Cloudera Navigator console after the external system has been successfully configured for use by Cloudera Navigator. See Authentication for Cloudera Navigator for configuration details.

Assigning User Roles to Groups

Cloudera Navigator user roles determine the Cloudera Navigator features and functions available to the logged in account holders. Data stewards, auditors, and other business users log in to the Cloudera Navigator console and have available only those menu selections and options appropriate for the user role (or roles) granted to group of which they are a member. The groups are defined in an external LDAP service that has been configured for use by Cloudera Navigator. See Authentication for Cloudera Navigator for details.

For example, a user belonging to a group granted only the Policy Editor role is limited to the Search, Analytics (metadata), and Policies tabs of the Cloudera Navigator console and to the features and functions available from those menus. See User Roles and Privileges Reference on page 174 for details about each Cloudera Navigator user role. Assigning or editing user roles requires logging in to the Cloudera Navigator console with an account having one of the following roles:

- Cloudera Manager Full Administrator or Navigator Administrator, or Cloudera Navigator User Administrator.

To assign Cloudera Navigator user roles to a group:

1. Log in to Cloudera Navigator console.
2. Click the Administration menu.
3. Click the Role Management tab.
4. Select Search for groups to search among all groups in the external directory.
   
   The Groups with Navigator roles choice shows groups that have already been assigned one or more Cloudera Navigator user roles.

5. Enter the name of the specific group in the search field.
6. Select the group from among those returned in the list.

   The details list any existing roles associated with this group. For example:
Figure 7: Group to Role Mapping

7. Click **Manage Role Assignment** in the upper right corner.
8. Select each user role you want to assign to the group.
9. Click **Save**.

Changes to user role assignments take effect the next time users in the group log in to Cloudera Navigator.

Displaying Roles for Current User Account Login

From the Cloudera Navigator console, you can verify the user roles associated with your current login by selecting **My Roles** from the account drop-down menu.

Managing Metadata Storage with Purge

The volume of metadata maintained by Navigator Metadata Server can grow quickly and exceed the capacity of the Solr instance that processes the index, which can affect search results speed and time to display data lineage. In addition, stale metadata may show relationships that no longer exist, or the lineage may take longer to display than necessary as the system processes extraneous details.

Cloudera Navigator's purge function removes metadata for files that have been deleted or for operations that are older than the specified timeframe. The result is faster search and more precise (up-to-date) lineage diagrams.

**Note:** Metadata for deleted files are only purged when the containing directory is also deleted.
In addition, purging before upgrading Cloudera Navigator to a new release can speed-up the upgrade process and reduce the chance of out-of-memory errors.

The purge function can be used in different ways:

- After deleting metadata, as detailed in Purging a Property.
- By using the Cloudera Navigator APIs, as detailed in Using the Purge APIs for Metadata Maintenance Tasks.
- For Cloudera Navigator 2.11 (and higher) releases, by using the Cloudera Navigator console and scheduling a regular weekly Purge, as detailed below.

Scheduling the Purge Process

Use the Cloudera Navigator console to configure a schedule for a regular weekly purge of deleted and stale metadata from the Navigator Metadata Server and its associated database.

**Note:** Cloudera recommends scheduling the purge process for non-production hours because other users and processes will be unable to use Cloudera Navigator until the purge process completes.

To configure the automated purge schedule:

1. [Log in to the Cloudera Navigator console](#) using an account with Full Administrator privileges.
2. Go to **Administration > Purge Settings** tab.

The current Metadata and Lineage purge schedule displays, along with lists of up to five upcoming scheduled purges and a list of up to five most recent completed purges.

To change the existing schedule:

1. Click **Edit**.
2. Set the purge process options.

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Range of selectable values and usage note</th>
</tr>
</thead>
<tbody>
<tr>
<td>How often</td>
<td>Weekly</td>
<td>Not configurable. The purge runs weekly per your specifications for Day and Time. It is enabled by default.</td>
</tr>
<tr>
<td>Day</td>
<td>Saturday</td>
<td>Select a day for the purge that will have minimal impact to your user community.</td>
</tr>
</tbody>
</table>
### Range of selectable values and usage note

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Range of selectable values and usage note</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time</strong></td>
<td>12 Midnight</td>
<td>Hourly time, from 12 Midnight through 11 PM. Select a time that will have minimal impact on production.</td>
</tr>
<tr>
<td><strong>Maximum purge duration</strong></td>
<td>12 hours</td>
<td>Set the amount of time you want to allow for the purge process to run. If not already complete, the HDFS purge process will not add any new items to purge after your specified duration. Entities purged to that point remain purged. All non-HDFS purge processes will run without limit. If set to 0, the purge is disabled. No other Cloudera Navigator operations, including through the console, can occur during the purge process.</td>
</tr>
<tr>
<td><strong>Purge HDFS entities deleted more than</strong></td>
<td>60 days</td>
<td>The number of days after an entity is deleted that elapse until the purge process removes its metadata. For example, a setting of 1 day purges entities deleted before two days ago but retains entities deleted yesterday.</td>
</tr>
<tr>
<td><strong>Purge SELECT operations</strong></td>
<td>Enabled</td>
<td>Hive and Impala SELECT operations older than days specified in <strong>Only Purge SELECT operations older than</strong> will be purged.</td>
</tr>
<tr>
<td><strong>Purge operations older than</strong></td>
<td>60 days</td>
<td>Yarn, Sqoop, and Pig operations older than the specified date will be purged. If <strong>Purge SELECT Operations</strong> is enabled, Hive and Impala SELECT operations older than the specified date will also be purged.</td>
</tr>
</tbody>
</table>

3. Click **Save** when finished.

Here is an example of a revised schedule:

![Example Schedule]

**Note:** If no schedules appear after you configure the purge process, your environment may not have automated purge enabled. To check if the automated purge is enabled, add `/debug` to the Navigator URL (http://fqdn-a.example.com:7187/debug), click **Configuration**, and check the value of `nav.purge.enabled`. If the property is set to false, you can enabled it in the debug page for this Navigator session; change the property in Cloudera Manager by adding `nav.purge.enabled=true` to the **Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera.navigator.properties**.

### What Metadata is Purged?

Purge processes look for metadata that is associated with deleted files and tables and with operation executions that are older than the configured threshold date.
Hive Metadata

- Hive operations
  - That don’t produce output
  - That have all operation executions that were executed earlier than the threshold date
- Hive operation executions
  - Associated with Hive operations that don't produce output
  - That were executed earlier than the threshold date
- Hive sub-operations
  - Associated with Hive operations that were purged
- All relations associated with the purged entities

Impala Metadata

- Impala operations
  - That don’t produce output
  - That have all operation executions that were executed earlier than the threshold date
- Impala operation executions
  - Associated with Impala operations that don't produce output
  - That were executed earlier than the threshold date
- Impala sub-operations
  - Associated with Impala operations that were purged
- All relations associated with the purged entities

Sqoop Metadata

- Sqoop import and export operations
  - That have all operation executions that were executed earlier than the threshold date and no existing
downstream entities
- Sqoop operation executions
  - that were executed earlier than the threshold date
- All relations associated with the purged entities

YARN Metadata

- YARN operations
  - That have all operation executions that were executed earlier than the threshold date
- YARN operation executions
  - That were executed earlier than the threshold date
- All relations associated with the purged entities

Pig Metadata

- Pig operations
  - That have all operation executions that were executed earlier than the threshold date and don’t apply to
tables connected to existing HDFS files
- Pig operation executions
That were executed earlier than the threshold date

- Pig tables
  - That were created by an operation execution executed earlier than the threshold date and also not connected to an existing HDFS file

- Pig fields
  - Fields in purged tables

- All relations associated with the purged entities

**HDFS Metadata**

- HDFS directories
  - That have been deleted longer than the configured threshold
  - AND don't have a logical-physical relation with another entity (such as a Hive table)
  - AND don't have children (sub directories or files) that aren't ready to be purged
  - AND aren't an endpoint in a data flow relation

- HDFS files
  - Deleted file metadata are purged only when the containing directory is purged

- All relations that have both endpoints associated with purged entities
Cloudera Navigator runs as two discrete roles—Navigator Audit Server and Navigator Metadata Server—in the context of Cloudera Manager Server. Features and configuration options available for these server-side processes are managed using the Cloudera Manager Admin Console. Management tasks include configuring filters to fine-tune audit event collection, tuning the Navigator Metadata Server for optimal performance, and configuring auditing for select services.

Management tasks also include implementing a backup system for Navigator metadata, and configuring security features, such as TLS/SSL to support HTTPS and client-server encrypted communications over the network.

Topics in this section apply to Navigator in general or to either Navigator Audit Server or Navigator Metadata Server specifically.

Accessing Navigator Data Management Logs

You can access Navigator Metadata Server and Navigator Audit Server logs from Cloudera Manager or from the command line on the host where each server is running.

Accessing Navigator Logs from Cloudera Manager

To access Navigator Metadata Server Logs from Cloudera Manager:

1. Select Clusters > Cloudera Management Service.
2. In the Status Summary, click Navigator Audit Server or Navigator Metadata Server.
3. Choose Log Files > Role Log File.

A new browser tab opens with a page of the most recent log. This example shows the audit server log; the metadata server log behaves the same.

Accessing Navigator Logs from the Command Line

By default, the Navigator server logs are located in /var/log on the host where the server is running. The host name is part of the log file name:

**Audit Server**
/var/log/cloudera-scm-navigator/mgmt-cmf-mgmt-NAVIGATOR-host.log.out

**Metadata Server**
/var/log/cloudera-scm-navigator/mgmt-cmf-mgmt-NAVIGATORMETASERVER-host.log.out
Back up Cloudera Navigator data

Backups of Cloudera Manager include the data stored for Navigator Audit Server and Navigator Metadata Server data, as described in Backing up Cloudera Manager. Follow these steps if you need to back up either storage independently from Cloudera Manager:

Backup Navigator Audit Server Data

Navigator Audit Server stores audit information in a database instance. The audit metadata does not need to be backed up at the same time as the Navigator Metadata Server metadata.

1. Confirm the Navigator Audit Server database location and connection information.

   To locate this information, log in to the Cloudera Manager Admin Console and go to Clusters > Cloudera Management Service. Click the Configuration tab. Select Scope > Navigator Audit Server and Category > Database. The results indicate the type, host, database name, and login account.

2. Using the appropriate tool for the database instance (such as MySQL), backup the database. See Backing up Cloudera Manager.

Backup Navigator Metadata Server Data

Navigator Metadata Server stores metadata in an embedded Solr data directory and in a database instance. Back up these locations together so the backups describe the same metadata state.

1. Shut down the Navigator Metadata Server role.

2. Backup the data directory.

   The Navigator Metadata Server default directory is /var/lib/cloudera-scm-navigator. To confirm that this is the path your deployment uses, log in to the Cloudera Manager Admin Console and search for the Navigator Metadata Server Storage Dir property (nav.data.dir).

3. Backup the database instance.

   • Confirm the Navigator Metadata Server database location and connection information.

      To locate this information, log in to the Cloudera Manager Admin Console and go to Clusters > Cloudera Management Service. Click the Configuration tab. Select Scope > Navigator Metadata Server. Select Category > Database. The results indicate the type, host, database name, and login account.

      • Using the appropriate tool for the database instance (such as MySQL), backup the database. See Backing up Cloudera Manager.

4. Start the Navigator Metadata Server role.

Authentication and Authorization

To support its user role-based authorization scheme, Cloudera Navigator integrates with external authentication mechanisms. External authentication mechanisms include:

• LDAP-compliant identity/authentication services, such as Active Directory and OpenLDAP
• SAML-based SSO solutions, such as Shibboleth and SiteMinder

Cloudera Manager Server has its own internal authentication mechanism, a database repository of user accounts. However, the user accounts defined in the internal Cloudera Manager account repository cannot be assigned Cloudera Navigator user roles. The only Cloudera Navigator user role that can be effectively applied to an account created in the Cloudera Manager internal repository is that of Navigator Administrator.

In other words, assigning Cloudera Navigator user roles to user accounts requires using an external authentication mechanism, such as Active Directory or OpenLDAP. See the following topics in the Cloudera Security guide for details:
Configuring Cloudera Navigator to work with Hue HA

Cloudera Navigator Search results include links to open data assets in Hue. No special configuration is required to enable these links.

However, for multiple Hue instances deployed behind external load-balancers such as in certain Hue high availability configurations, the links are properly generated only if the host name of the external load-balancer is set as the preferred URL, by configuring the Advanced Configuration Property (Safety Valve) as follows:

1. Log in to Cloudera Manager Admin Console using an account with Full Administrator privileges.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. Under the Category filter, click Advanced.
6. Scroll to the property Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties.
7. Replace the cluster name and set the nav.hue.preferred_baseurl.<clusterName> to the fully-qualified domain name (FQDN) or specific URL of your Hue load-balancer using one of the following formats:

```
nav.hue.preferred_baseurl.<clusterName>=<FQDN>
```

or

```
nav.hue.preferred_baseurl.<clusterName>=https://<FQDN>:<port>
```

Use the full URL format when Hue is configured to use TLS/SSL; indicate https and the secure port number for the load balancer.

For example, if a load-balancer for multiple Hue services is setup on host hue_lb_nginx.example.com and configured for TLS/SSL on port 8889, the entry would be as follows:

```
nav.hue.preferred_baseurl.myCluster=https://hue_lb.example.com:8889
```

8. Click Save.
9. Restart the Navigator Metadata Server role.

Links to Hue will now be properly created and available in the Search results lists in the Cloudera Navigator console.
Encryption (TLS/SSL) and Cloudera Navigator

Cloudera Navigator supports TLS/SSL encryption for network communications between the Navigator Metadata Server and clients, such as the web browser used for Cloudera Navigator console, and between RPC client and server processes for Navigator Metadata Server and Navigator Audit Server. See Cloudera Security for configuration details, specifically:

- Configuring TLS/SSL for Cloudera Navigator Audit Server
- Configuring TLS/SSL for Cloudera Navigator Metadata Server

Limiting Sensitive Data in Navigator Logs

Cloudera Navigator collects metadata for cluster activities and assets; typically, this information does not include data and therefore doesn’t provide an opportunity to expose sensitive data through the user interface. However, there could be sensitive data exposed the Navigator logs: queries from Impala or Hive may include a specific data value, such as a phone number or US social security number.

To guard against sensitive data exposures through Navigator logs, you can use Cloudera Manager's log redaction feature to filter out string patterns using regular expressions. For example, you could mask all quoted strings, social security numbers, and phone numbers.

The log redaction rules can apply across all cluster services.

For details, see the Cloudera Security guide, specifically Log and Query Redaction.

Preventing Concurrent Logins from the Same User

You can configure a limit to the number of simultaneous sessions authenticated against the same user name. The property `nav.max.concurrent.sessions` takes an integer value as a limit; set the value to -1 (default) to turn off the limit. To change the default behavior, in Cloudera Manager, add the property with a new value to the Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for `cloudera-navigator.properties`. Restart the Navigator Metadata Server to apply the change.

Navigator Audit Server Management

The Navigator Audit Server tracks and coalesces events from Cloudera Manager and stores them in the Navigator Audit database. This section provides a high level view of the auditing architecture and shows administrators how to use Cloudera Manager Admin Console to add the Navigator Audit Server to an existing cluster, how to configure some of its features.

Navigator Auditing Architecture

The figure below shows a high level view of the Cloudera Navigator auditing architecture:
During system setup, plug-ins for the various services—HDFS, HBase, and Hive (HiveServer2, Beeswax servers) services, for example—are enabled. These plug-ins work with the service to collect and filter events emitted by the respective service, writing the events to an audit log on the local filesystem. Impala, Sentry, and the Cloudera Navigator Metadata Server also collect and filter events and write them to their respective audit log files.

**Note:** When upgrading Cloudera Navigator, always check the requirements for any preliminary tasks or constraints due to service plug-ins.

**Auditing Architecture In More Detail: How It Works**

Here is some more detail about the auditing architecture and interaction among Cloudera Manager Agent, local log file, and Navigator Audit Server.

The Cloudera Manager Agent process on each host in the cluster:

- Monitors local audit log files
- Sends events captured in the logs to the Navigator Audit Server
- Retries sending any event that fails to transmit successfully
- Keeps track of successfully transmitted events from the logfile (offset position in the file) to prevent re-sending any already processed events after a system failure and restart
- Deletes old audit logs after successful transmission to the Navigator Audit Server

After any event is written to the audit log file (and assuming space available on the filesystem), its delivery is guaranteed. In other words, transient (in-memory) buffer handling is not involved in this part of the process. Audit logs are rotated and the Cloudera Manager Agent follows the rotation of the log.

The plug-in for each of the various services effectively writes the events to the audit log file. A plug-in that fails to write an event to the audit log file can either drop the event or can shut down the process in which it is running depending on the configured queue policy.

The Navigator Audit Server performs the following functions:
• Tracks and coalesces events obtained from Cloudera Manager
• Stores events to the Navigator Audit database

Auditing can be enabled for the core services listed in the table. Use the Cloudera Manager Admin Console to change the configuration details and to create whitelists and blacklists for better control over audit details collected by Navigator Audit Server. See Configuring Audit and Log Properties for details.

<table>
<thead>
<tr>
<th>Audited Service</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>HBase</td>
<td>hbase-MASTER, hbase-REGIONSERVER</td>
</tr>
<tr>
<td>HDFS</td>
<td>hdfs-NAMENODE</td>
</tr>
<tr>
<td>Hive</td>
<td>Hive Server 2 only (hive-HIVESERVER2). Command line beeline (not hive).</td>
</tr>
<tr>
<td>Hue</td>
<td>impala-IMPALAD (Auditing data and lineage data both collected by Cloudera Manager Agent).</td>
</tr>
<tr>
<td>Navigator Metadata Server</td>
<td></td>
</tr>
<tr>
<td>Sentry</td>
<td></td>
</tr>
<tr>
<td>Solr</td>
<td></td>
</tr>
</tbody>
</table>

Setting Up Navigator Audit Server

The steps below show you how to add the Navigator Audit Server role to an existing Cloudera Manager cluster.

The Navigator Audit Server role runs on the Cloudera Management Service. By default, the installation process installs both Navigator Audit Server and Navigator Metadata Server roles on the same Cloudera Management Service instance, but that may not be optimal, especially for very large clusters.

Adding the Navigator Audit Server Role

**Cloudera Manager Required Role:** Navigator Administrator (or Full Administrator)

The steps below assume that an external database is running and available to be used with the Navigator Audit Server role. Before adding this role, gather the configuration details about the external database instance so you can enter them when needed during this process.

To add the Navigator Audit Server role to the cluster:

1. Log in to the Cloudera Manager Admin Console.
2. From the menu, select Clusters > Cloudera Management Service.
3. Click the Instances tab.
4. Click Add Role Instances. The Customize Role Assignments page displays and lists the nodes available to support additional roles, assuming the cluster has available resources. Reassign and customize role instances as needed.
5. Select the Navigator Audit Server role and assign it to appropriate host for your cluster.
6. When finished, click Continue. The Database Setup page displays.
7. Click Use Custom Database.
8. In the Navigator Audit Server section, enter the details of your database instance:
   • Database host name
   • Database type
   • Database name
   • Username
   • Password
Note: For non-production use only, you can select Use Embedded Database. Cloudera Manager creates and configures the databases for you. Make a note of the generated passwords.

Here is an example of a configured external database instance:

<table>
<thead>
<tr>
<th>Navigator Audit Server Database Type</th>
<th>Navigator Audit Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigator.db.type</td>
<td>MySQL</td>
</tr>
<tr>
<td></td>
<td>Oracle</td>
</tr>
<tr>
<td></td>
<td>PostgreSQL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Navigator Audit Server Database Hostname</th>
<th>Navigator Audit Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigator.db.host</td>
<td></td>
</tr>
<tr>
<td>freqn-1.example.com:3306</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Navigator Audit Server Database Name</th>
<th>Navigator Audit Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigator.db.name</td>
<td></td>
</tr>
<tr>
<td>nas-db</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Navigator Audit Server Database Username</th>
<th>Navigator Audit Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigator.db.user</td>
<td></td>
</tr>
<tr>
<td>audit</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Navigator Audit Server Database Password</th>
<th>Navigator Audit Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigator.db.password</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td></td>
</tr>
</tbody>
</table>

9. Click Test Connection to verify the communication between the cluster and the external database. If the test fails, check the database settings and try again. If you selected embedded database, a message displays regarding database creation.
10. Click Continue.
12. Click Finish.

Starting, Stopping, and Restarting the Navigator Audit Server

1. Log in to the Cloudera Manager Admin Console.
2. From the menu, select Clusters > Cloudera Management Service.
3. Click the Instances tab.
4. Click the link for the Navigator Audit Server from the Role Type list. The Actions for Selected button activates and displays (1) for the selected role.
5. From the Actions for Selected (1) menu, select the Action you want to perform on this role instance:
   - Start
   - Stop
   - Restart
   - Enter Maintenance Mode
   - Exit Maintenance Mode
   - Delete

   A confirmation message displays prompting you to Cancel or complete the specified Action.
6. Click Cancel to abandon the process, or click Action to execute the action.
Configuring the Navigator Audit Server Data Expiration Period

By default, the Navigator Audit Server keeps 90 days worth of audit events in its database. The setting can be changed to a shorter timeframe by configuring the number of hours or days of audit events to keep in the Navigator Audit Server database as follows:

1. Log in to the Cloudera Manager Admin Console.
2. From the menu, select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Select Navigator Audit Server from the Scope filter.
5. For the Navigator Audit Server Data Expiration Period, enter the number of days or hours or audit events that should be retained in the database before purging and select either day(s) or hour(s) from the drop-down to label the value accordingly.
6. Restart the Navigator Audit Server role.

Review the Navigator Audit Server Filters

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

By far, HDFS produces the largest volume of audit events. Many of these events are caught and discarded by audit filters, particularly events that don’t uniquely describe activity on the cluster or are produced only by controlled service users. Many of the default HDFS audit filters assume default service role names. To get value from the default filters, be sure to review them to make sure that the role names referenced match those used in your environment.

See Default and Recommended Audit Filters for HDFS for more information about the provided filters and their use.

You can review and possibly change the filters as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select an HDFS service. Select Cluster > HDFS Service-Name
3. Click the Configuration tab.
4. Filter the properties on "audit".
5. Review the list of filters in the property Audit Event Filter.

- Action: accept Fields: allowed: (?::false)
  Collect events where the value of the audit event field "allowed" is false. This filter is first in the list to make sure that all events are tested against this filter.

- Action: discard Fields: src:
  (?:.*\./.\.hive-staging($|.*))?.*\./.\.staging($|.*))?.*\./.\.sparkStaging($|.*))?.*\./.\impala_insert_staging($|.*))?\./user/history\done_intermediate(?::.*)?\./user/spark/spark2ApplicationHistory($|.*))?\./user/spark/applicationHistory($|.*))?\./user/hue/\\.cloudera_manager_hive_metastore_canary(?::.*)?\./user/hue/\\.Trash/\Current\user/hue/\\.cloudera_manager_hive_metastore_canary(?::.*)?\./tmp(?::.*)?\.
  Don’t collect events that occur in these directories. Typically, HDFS operations in these directories represent background operations performed by service users. Consider modifying the paths to match how your environment is configured or removing directories from the list. Note that the regular expression entry ($|.*)) indicates the directory itself and any subdirectories.

- Action: accept Fields: operation: delete|rename.*
  Collect events that describe HDFS delete or rename operations. The regular expression entry .* indicates that add types of rename events are collected, such as rename options=1.

- Action: discard Fields: username:
  (?::cloudera-scm|dr.who|hbase|hive|impala|mapred|solr|spark)(?::.*)?
  Collect events that describe HDFS delete or rename operations. The regular expression entry .* indicates that add types of rename events are collected, such as rename options=1.
Don’t collect operations performed by this list of service users. Note that these are the default role names suggested by Cloudera Manager. If your environment uses other role names, consider replacing the names in the filter.

- **Action**: discard
  - **Fields**: `username: (?::hdfs)(?::/+)?, operation: (?::listStatus|listCachePools|listCacheDirectives|getfileinfo)`

Don’t collect these specific operations performed by the HDFS superuser. These operations describe metadata operations that do not reveal data access attempts. In addition, these operations tend to be background operations that are triggered when other services perform operations. Removing them from the audit log significantly reduces the volume of audit events without losing critical events. Note that this filter uses `hdfs` as the HDFS superuser account. Your environment may be configured with a different or more than one different user name with superuser privileges. To make this filter effective, change the name to that used in your system.

- **Action**: accept
  - **Fields**: `operation: (?::getfileinfo)`

Added as a placeholder. Switch the action to discard to enable the filter. When set to discard, this filter does not collect any HDFS `getfileinfo` operation events. This is recommended for systems where this metadata event is not a meaningful audit event. Discarding it can make a significant reduction in the overall volume of the audit database as the `getfileinfo` event occurs along with every other action in HDFS directly or through services that access HDFS. Left as it is (accept), it has no effect.

6. Make changes to the filters as appropriate.
7. **Restart the Navigator Audit Server role.**

### Setting the Navigator Audit Server Java Heap Size

**Cloudera Manager Required Role:** [Navigator Administrator](#) (or [Full Administrator](#))

The Navigator Audit Server performance is typically bound by the performance of the database. With that in mind, consider increasing the Navigator Audit Server Java heap to as much as 4 GB; increasing heap beyond this point is unlikely to change the server performance. The default Java heap is set to 1.5 GiB.

You can change the setting as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select **Clusters > Cloudera Management Service.**
3. Click the **Configuration** tab.
4. Select **Scope > Navigator Audit Server.**
5. For **Filter Category**, click **Resource Management** to display the Java Heap Size property.

   ![Java Heap Size of Auditing Server in Bytes](#)

   **Navigator Audit Server Default Group**

   - 4 GiB

6. Use the drop-down selector to change the unit scale to B (Bytes), KiB, MiB, or GiB and set your preferred heap size.
7. Click **Save Changes.**

   The setting takes effect only after restarting the role. Restart the Navigator Metadata Server role now or make other configuration changes and restart after you are finished with all changes.

8. **Restart the Navigator Audit Server role.**

### Configuring the Navigator Audit Server Log Directory

The default location for the Navigator Audit Server logs is:

```
/var/log/cloudera-scm-navigator
```
To change the location for the log directory:

1. Log in to the Cloudera Manager Admin Console.
2. From the menu, select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Select Category > Logs.
5. Enter the path for the Navigator Audit Server Log Directory property.
6. Restart the Navigator Audit Server role.

Enabling Audit and Log Collection for Services

**Cloudera Manager Required Role:** Navigator Administrator (or Full Administrator)

Auditing of every service and role in the cluster may not be necessary and may degrade performance, which is why auditing is not always enabled by default. In addition, auditing can be configured to capture only specific events, as detailed in Configuring Service Auditing Properties.

### Enabling Audit Collection

Any service or role instance that can be audited by Cloudera Navigator has an Enable Audit Collection property. When enabled, the Cloudera Manager Agent process on the node monitors the audit log file (or files) and sends collected audit records to the Navigator Audit Server.

1. Log in to Cloudera Manager Admin Console
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Select ServiceName (Service-Wide) for the Scope filter.
5. Select Navigator Metadata Server for the Category filter.
6. Click the Enable Audit Collection checkbox to activate auditing for the service.
7. Click Save Changes.
8. Restart the service.

### Configuring Impala Daemon Logging

To enable logging for the Impala Daemon role.

1. Log in to Cloudera Manager Admin Console
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Select Impala Daemon for the Scope filter.
5. Select Logs for the Category filter.
6. Edit the Enable Impala Audit Event Generation.
7. Click Save Changes.
8. Restart the Impala daemon.

To set the log file size:

1. Click the Impala service.
2. Select Scope > Impala Daemon.
3. Select Category > Logs.
4. Set the Impala Daemon Maximum Audit Log File Size property.
5. Click Save Changes.
6. Restart the Impala service.

### Enabling Solr Auditing

Solr auditing is disabled by default. To enable auditing:

1. Enable Sentry authorization for Solr following the procedure in Enabling Sentry Policy File Authorization for Solr.
2. Go to the Solr service.
3. Click the Configuration tab.
4. Select Scope > Solr Service (Service-Wide)
5. Select Category > Policy File Based Sentry category.
6. Select or clear the Enable Sentry Authorization checkbox.
7. Select Category > Cloudera Navigator category.
8. Select or clear the Enable Audit Collection checkbox. See Configuring Service Audit Collection and Log Properties.
9. Click Save Changes to commit the changes.
10. Restart the service.

Configuring Audit Logs

The following properties apply to an audit log file:

- **Audit Log Directory** - The directory in which audit log files are written. By default, this property is not set if Cloudera Navigator is not installed.

  A validation check is performed for all lifecycle actions (stop/start/restart). If the Enable Collection flag is selected and the Audit Log Directory property is not set, the validator displays a message that says that the Audit Log Directory property must be set to enable auditing.

  If the value of this property is changed, and service is restarted, then the Cloudera Manager Agent will start monitoring the new log directory for audit events. In this case it is possible that not all events are published from the old audit log directory. To avoid loss of audit events, when this property is changed, perform the following steps:

  1. Stop the service.
  2. Copy audit log files and (for Impala only) the impalad_audit_wal file from the old audit log directory to the new audit log directory. This needs to be done on all the hosts where Impala Daemons are running.
  3. Start the service.

- **Maximum Audit Log File Size** - The maximum size of the audit log file before a new file is created. The unit of the file size is service dependent:

  - HDFS, HBase, Hive, Hue, Navigator Metadata Server, Sentry, Solr - MiB
  - Impala - lines (queries)

- **Number of Audit Logs to Retain** - Maximum number of rolled over audit logs to retain. The logs will not be deleted if they contain audit events that have not yet been propagated to the Audit Server.

To configure audit logs do the following:

1. Do one of the following:
   - Service - Go to a supported service.
   - Navigator Metadata Server
     - Do one of the following:
       - Select Clusters > Cloudera Management Service.
       - On the Home > Status tab, in Cloudera Management Service table, click the Cloudera Management Service link.

2. Click the Configuration tab.
3. Select the scope according to the service:
   - All services except Impala - Scope > ServiceName (Service-Wide).
   - Impala - Scope > Impala Daemon.
   - Navigator Metadata Server - Scope > Navigator Metadata Server.
4. Select Category > Logs.
5. Configure the log properties. For Impala, preface each log property with **Impala Daemon**.
6. Click **Save Changes**.
7. Restart the service.

### Configuring Audit Properties

Set auditing properties for a service in the service’s configuration settings in Cloudera Manager.

**Cloudera Manager Required Role:** [Navigator Administrator](#) (or **Full Administrator**)

#### Setting Audit Properties

To review or update audit properties for a service:

1. Log in to the Cloudera Manager Admin Console.
2. Select select any service that supports auditing. Select **Cluster > Service-Name**
3. Click the **Configuration** tab.
4. Select **Service** (Service-Wide) for the **Scope** filter.
5. Select **Cloudera Navigator** for the **Category** filter.
6. Edit the properties.
7. Click **Save Changes** to commit the changes.
8. Restart the service if needed.

   It is not necessary to restart the service if you are updating an audit filter.

In addition, for Impala and Solr auditing, perform the steps in **Configuring Impala Daemon Logging** on page 93 and **Enabling Solr Auditing** on page 93 and for all services perform the steps in **Configuring Audit Logs** on page 94.

Operations within Navigator are also audited. To update Navigator’s audits:

1. Select **Clusters > Cloudera Management Service**.
2. Click the **Configuration** tab.
3. Select **Navigator Metadata Server** for the **Scope** filter.
4. Select **Category > Cloudera Navigator** category.
5. Edit the properties.
6. Click **Save Changes** to commit the changes.
7. Restart the service if needed.

   It is not necessary to restart the service if you are updating an audit filter.

#### Service-level Audit Properties

Each service (with exceptions noted) that supports auditing has the following properties:

- **Enable Audit Collection** - See **Enabling Audit Collection** on page 93 and **Enabling Solr Auditing** on page 93.
- **Audit Log properties** - See **Configuring Audit Logs** on page 94 and **Configuring Impala Daemon Logging**.
- **Audit Event Filter** - A set of rules that capture properties of auditable events and actions to be performed when an event matches those properties. The Cloudera Manager Agent uses this property to filter events out *before* they are sent to Cloudera Navigator.
- **Audit Event Tracker** - A set of rules for tracking and coalescing events. This feature is used to define equivalency between different audit events. Tracking works by keeping a reference to events when they first appear, and comparing other incoming events against the tracked events according to the rules defined. When events match, according to a set of configurable parameters, only one entry in the audit list is generated for all the matching events. This property is not supported for the Navigator Metadata Server.
- **Audit Queue Policy** - The action to take when the audit event queue is full. The options are Drop or Shutdown. When a queue is full and the queue policy of the service is Shutdown, before shutting down the service, *N* audits will be discarded, where *N* is the size of the Cloudera Navigator Audit Server queue.
**Note:** If the queue policy is Shutdown, the Impala service is shut down only if Impala is unable to write to the audit log file. It is possible that an event may not appear in the audit event log due to an error in transfer to the Cloudera Manager Agent or database. In such cases Impala will not shut down and will keep writing to the log file. When the transfer problem is fixed the events will be transferred to the database.

This property is not supported for Hue or the Navigator Metadata Server.

**Finding Audit Filter Properties for Each Service**

For information on the structure of the objects, and the properties for which you can set filters, display the description on the configuration page as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select any service that supports auditing. Select Cluster > **Service-Name**
3. Click the Configuration tab.
4. Select **Service** (Service-Wide) for the Scope filter.
5. Select Cloudera Navigator for the Category filter.
6. Click for Audit Event Tracker to open the property description.

For example, the Hive properties are:

- operation: the Hive operation being performed.
- userName: the user performing the action.
- ipAddress: the IP from where the request originated.
- allowed: whether the operation was allowed or denied.
- databaseName: the databaseName for the operation.
- tableName: the tableName for the operation.
- objectType: the type of object affected by the operation.
- resourcePath: the path of the resource affected by the operation.

**Entering Audit Filter Properties**

The Audit Event Filter and Audit Event Tracker rules for filtering and coalescing events are expressed as JSON objects. You can edit these rules using a rule editor. The `username` filters shown here include an optional phrase \((?:/.+)?\) to filter events where the user name is constructed with a slash as part of the name, such as a Kerberos principal \((username/instance@REALM)\). It applies only if the regular expression matches one of the usernames listed in the first phrase. The `src` filter shows a similar optional phrase \((?:/.*\)?\) that filters for subdirectories under the `/tmp` directory. The regular expression looks for occurrences of slash followed by zero or more characters; this makes sure that subdirectories are included but other directories whose name starts with "tmp" are not included in the filter.

![Audit Filter Properties](image)

or in a JSON text field:
Default and Recommended Audit Filters

The default filter settings are set for the audited service:

**HBase**

Default filters discard events that affect the `-ROOT-`, `.META.`, and `acl` tables.

**Hive**

Default filters discard events generated by Hive MapReduce jobs in the `/tmp` directory.

**Impala, Solr, Navigator Metadata Server**

There are no default filters set for these services.

**HDFS**

Starting in Cloudera Manager version 5.16.0, the default HDFS audit filters were changed to be more precise about collecting meaningful audit events. If your system included customized audit filters, the upgrade to this version did not apply the new filters. Please review the following filters and consider adding them to your audit filters. Note that the order of the filters is important: the first filter is applied to incoming events first. The recommended HDFS audit
Filters are described here and provided in a JSON version that can be customized and pasted into the Cloudera Manager property for HDFS audit filters.

- **Action:** accept, **Fields:** allowed: (?i:false)
  Collect events where the value of the audit event field "allowed" is false. This filter is first in the list to make sure that all events are tested against this filter.

- **Action:** discard, **Fields:**
  src:
  (?i:^/.*\/.*/\.*$)|.*\/(\.*\/.*)?
  Don’t collect events that occur in these directories. Typically, HDFS operations in these directories represent background operations performed by service users. Consider modifying the paths to match how your environment is configured or removing directories from the list. Note that the regular expression entry ($|.*$) indicates the directory itself and any subdirectories.

- **Action:** accept, **Fields:**
  operation: delete|rename.*
  Collect events that describe HDFS delete or rename operations. The regular expression entry . * indicates that add types of rename events are collected, such as rename options=1.

- **Action:** discard, **Fields:**
  username: (?i:cloudera-scm|dr.who|hbase|hive|impala|mapred|solr|spark)(?i:/.*)?
  Don’t collect operations performed by this list of service users. Note that these are the default role names suggested by Cloudera Manager. If your environment uses other role names, consider replacing the names in the filter.

- **Action:** discard, **Fields:**
  operation: (?i:listStatus|listCachePools|listCacheDirectives|getfileinfo)
  Username: (?i:hdfs)(?i:/.*)?,
  Don’t collect these specific operations performed by the HDFS superuser. These operations describe metadata operations that do not reveal data access attempts. In addition, these operations tend to be background operations that are triggered when other services perform operations. Removing them from the audit log significantly reduces the volume of audit events without losing critical events. Note that this filter uses hdfs as the HDFS superuser account. Your environment may be configured with a different or more than one different user name with superuser privileges. To make this filter effective, change the name to that used in your system.

- **Action:** accept, **Fields:**
  operation: (?i:getfileinfo)
  Added as a placeholder. Switch the action to discard to enable the filter. When set to discard, this filter does not collect any HDFS getfileinfo operation events. This is recommended for systems where this metadata event is not a meaningful audit event. Discarding it can make a significant reduction in the overall volume of the audit database as the getfileinfo event occurs along with every other action in HDFS directly or through services that access HDFS. Left as it is (accept), it has no effect.

**Audit Filter Examples**

Use audit filters to control what information is collected by Navigator Audit Server. The following sections describe cases where a filter can allow you to remove noise from the audit events or otherwise improve the quality of the audit trail.
Removing audit events for automated tasks

If you find that the audit events include references to automated tasks against HDFS or other services that you don't want to include in your audit, you can include the tasks in a "discard" filter. For example, to discard all tasks performed against HDFS by the user under which the automated tasks are performed, create an HDFS filter:

- **Action**: Discard
- **Fields**: username: (?:automationuser|automationuser@realm.example.com)

This rule would discard any event performed by the **automationuser** account, where the user is specified by itself or with a Kerberos realm designation. Note that the filter is not case sensitive. As a performance optimization, the expression is included in parentheses with `?:` to indicate that the matched values are not retained.

Removing audit events for low priority operations

You may find that your audit log includes reference to events that you are not interested in tracking. You can prevent these events from being included in the audit log by filtering them out by operation. For example, if the audit event that you don't want to track looks like this:

```
2170816,hdfs,1,someuser@MYREALM.COM,null,192.168.12.18,1477605498525,getfileinfo,/user/someuser/.staging/jobx/status,null,null,null
```

where the HDFS operation **getfileinfo** ran against the file **status** for the user **someuser** on a node in the cluster. The user account is indicated with a Kerberos realm.

To remove the HDFS getfileinfo accesses by this user, the audit filter would reference the user (by itself or with a Kerberos realm name) and the getfileinfo operation:

- **Action**: Discard
- **Fields**: username: (?:someuser|someuser@MYREALM.com)
  
  operation: (?:getfileinfo)

As a performance optimization, the expressions are included in parentheses with `?:` to indicate that the matched values are not retained.

To include more than one operation in the same filter, include both operations with an OR operator `|`:

- **Action**: Discard
- **Fields**: username: (?:someuser|someuser@MYREALM.com)
  
  operation: (?:getfileinfo|liststatus)

Monitoring Navigator Audit Service Health

Cloudera recommends monitoring the Navigator Audit Service to ensure that it is always running. This is especially important to ensure that complete and immutable audit records will be available when needed for corporate governance and compliance purposes.

The Navigator Audit Service has a self-check mechanism, the Audit Pipeline Health Check, that generates warning messages when the system slows down or fails. The health check determines failures and slow downs by keeping track of the number of bytes of audit data sent, bytes unsent, or failures during the send process that occur within the monitoring window. Healthy state for a given monitoring period results when all audit data is sent (no data remaining), with error-free transmission from agent to server.

The health check can be run for each service role instance (daemon) that can generate events. By default, all service role groups are selected when you enabled the health check, and the default monitoring period is 20 minutes. Thresholds for warning messages and the monitoring period are configurable using the Cloudera Manager Admin, as detailed below.

Configuring the Audit Pipeline Health Check

**Cloudera Manager Required Role**: Navigator Administrator (or Full Administrator)
Log in to the Cloudera Manager Admin Console. For each service that you want to enable the health check, follow the steps below.

1. Select Clusters > service-name.
2. Click the Configuration tab.
3. In the Search field, type mgmt.navigator to display the health check configuration properties for the service:

   ![Health Check Configuration Properties](image)

   4. Modify the settings to shorten (or lengthen) the monitoring period as needed for your system and set bytes for Warning and Critical thresholds.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigator Audit Pipeline Health Check</td>
<td>Check the box to enable the health check. Health check can be enabled for specific groups. By default, the health check is enabled for all groups.</td>
</tr>
<tr>
<td>Monitoring Period for Audit Failures</td>
<td>Default is 20 minutes. The amount of time to process audit events (count processed and evaluate other metrics before generating warnings.</td>
</tr>
<tr>
<td>Navigator Audit Failure Thresholds</td>
<td>Size (in bytes) of failures that will trigger messages.</td>
</tr>
<tr>
<td></td>
<td>• Warning—Default is Never. Set this to an amount of bytes and condition (which to trigger a warning message.</td>
</tr>
<tr>
<td></td>
<td>• Critical—Default is Any, meaning that any amount of failed audit data sent from Cloudera Manager to the server will trigger a critical message.</td>
</tr>
</tbody>
</table>

5. Click Save Changes.

For example, as shown in the Cloudera Manager Admin Console, the pipeline health check is enabled for all groups in the service. The failure period is set to 15 minutes, and the health check sends a warning for failures of any size and a critical error when 2 KiB of audit events have not been sent.
Publishing Audit Events

**Minimum Required Role:** Navigator Administrator (also provided by Full Administrator)

Audit events can be published to a Kafka topic or to the system log (syslog). After configuring Cloudera Navigator to send audit events, failures to send events are logged to the Audit Server log.

**Publishing Audit Events to Kafka**

Navigator can stream audit events to a Kafka topic so the events can be used by another application or archived outside of the Navigator Audit Server database. To configure Navigator to stream audits to a Kafka topic:

1. Create the topic in Kafka.
   - If the Kafka instance is configured with Topic Auto Creation enabled, you can skip this step. The topic will be created when the first event is sent to Kafka.
   - For steps for creating a Kafka topic, see the Kafka \texttt{kafka-topics} command in Using Apache Kafka Command-line Tools.

2. Log in to Cloudera Manager Admin Console using either Navigator Administrator or Full Administrator privileged account.


4. Click the Configuration tab.

5. If Kafka is managed by the same Cloudera Manager instance as Navigator:
   - In the Search field, type "kafka" to display the configurable settings.
   - For the Kafka Service property, select the Kafka service to which Cloudera Navigator will publish audit events.
   - For the Kafka Topic property, enter the name of the topic to which Cloudera Navigator will publish the audit events.

6. If Kafka is not managed by the same Cloudera Manager instance as Navigator, add properties to accommodate logging for Kafka:
   - In the Search field, type "logging" to display the configurable settings.
b. For the **Navigator Audit Server Logging Advanced Configuration Snippet (Safety Valve)** property, insert the following logging properties, inserting the `brokerList` and `topic` values with the values for your system:

```java
log4j.logger.kafkaAuditStream=TRACE,KAFKA
log4j.appender.KAFKA=org.apache.kafka.log4j.KafkaLog4jAppender
log4j.additivity.kafkaAuditStream=false
log4j.appender.KAFKA.layout=org.apache.log4j.PatternLayout
log4j.appender.KAFKA.layout.ConversionPattern=%m%n
log4j.appender.KAFKA.syncSend=false
log4j.appender.KAFKA.brokerList=<comma-separated list of brokers as host:port>
log4j.appender.KAFKA.topic=<topic>
```

If Kafka is managed using Cloudera Manager, you can find the broker host names in the **Instances** tab for the Kafka service. List the host and port numbers in a comma-separated list without spaces. For example:

```none
kafka-dedicated-1.example.com:9092,kafka-dedicated-2.example.com:9092,kafka-dedicated-3.example.com:9092
```

c. If Kafka is not managed by the same Cloudera Manager instance as Navigator and Kerberos is enabled in the Kafka cluster, there are additional configuration steps.

These steps are described in the Kafka client documentation for security,  [Configuring Apache Kafka Security](#).

- Include the following additional properties in the **Navigator Audit Server Logging Advanced Configuration Snippet (Safety Valve)**, inserting the truststore `location` and truststore `password` with the values from your system:

```java
log4j.appender.KAFKA.SaslKerberosServiceName=kafka
log4j.appender.KAFKA.clientJaasConfPath=navigator.jaas.conf
log4j.appender.KAFKA.SecurityProtocol=SASL_SSL
log4j.appender.KAFKA.SslTruststoreLocation=<path/to/truststore>
log4j.appender.KAFKA.SslTruststorePassword=<password>
```

- For the Navigator Kafka client create a `jaas.conf` file with keytabs.

To generate keytabs, see [Step 6: Get or Create a Kerberos Principal for Each User Account](#).

Insert the `keyTab` and `principal` with the values from your system.

```java
KafkaClient {
  com.sun.security.auth.module.Krb5LoginModule required
  useKeyTab=true
  keyTab="/etc/security/keytabs/mykafkaclient.keytab"
  principal="mykafkaclient/clients.hostname.com@EXAMPLE.COM";
};
```

- Set the location of the `jaas.conf` in the Navigator host environment.

```bash
export KAFKA_OPTS="-Djava.security.auth.login.config=/home/user/jaas.conf"
```

7. Add a description of the changes and click **Save Changes**.
8. Restart the role.
9. You can validate that events are published to the topic using the Kafka `kafka-console-consumer` command as described in [Using Apache Kafka Command-line Tools](#).

### Publishing Audit Events to Syslog

The Audit Server logs all audit records into a `Log4j` logger called `auditStream`. The log messages are logged at the `TRACE` level, with the attributes of the audit records. By default, the `auditStream` logger is inactive because the logger level is set to `FATAL`. It is also connected to a `NullAppender`, and does not forward to other appenders (additivity set to false).
To record the audit stream, configure the `auditStream` logger with the desired appender. For example, the standard `SyslogAppender` allows you to send the audit records to a remote syslog.

The Log4j `SyslogAppender` supports only UDP. An example syslog configuration would be:

```
$ModLoad imudp
$UDPServerRun 514
# Accept everything (even DEBUG messages)
local2.* /my/audit/trail.log
```

Other appenders can be attached to the `auditStream` to implement other integration behaviors.

Audit events published to syslog can be formatted as JSON or RSA EnVision. To configure audit logging to syslog:

1. Log in to Cloudera Manager Admin Console using either Navigator Administrator or Full Administrator privileged account.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Locate the **Navigator Audit Server Logging Advanced Configuration Snippet** property by typing its name in the Search box.
5. Depending on the format type, enter:

   ```
   log4j.appenders.SYSLOG = org.apache.log4j.net.SyslogAppender
   log4j.appenders.SYSLOG.SyslogHost = hostname
   log4j.appenders.SYSLOG.Facility = Local2
   log4j.appenders.SYSLOG.FacilityPrinting = true
   ```

   To configure the specific stream type, enter:

<table>
<thead>
<tr>
<th>Format</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON</td>
<td><code>log4j.logger.auditStream = TRACE,SYSLOG</code></td>
</tr>
<tr>
<td></td>
<td><code>log4j.additivity.auditStream = false</code></td>
</tr>
<tr>
<td>RSA EnVision</td>
<td><code>log4j.logger.auditStreamEnVision = TRACE,SYSLOG</code></td>
</tr>
<tr>
<td></td>
<td><code>log4j.additivity.auditStreamEnVision = false</code></td>
</tr>
</tbody>
</table>

6. Click **Save Changes**.
7. Restart the role.

**Maintaining Navigator Audit Server**

**Cloudera Manager Required Role:** [Navigator Administrator](#) (or [Full Administrator](#))

When you have Cloudera Navigator running, Navigator Audit Server is enabled by default. Audits are enabled for all supported services, including HDFS, Hive, Impala, Hue, HBase, Sentry, and Solr as well as Cloudera Manager and Navigator itself. Events are retained for 90 days with some default filters defined.

Because the default settings may not work for all environments, we recommend that you review your Navigator Audit Server setup to review these three areas:

**What events are collected?**

Default filters remove some service-to-service events. However, you should check to make sure that the role names used in the filters match what’s used in your system.

**How long are events retained?**

To determine the best expiration period for events in your audit system, consider the volume of incoming events, the disk space available for the audit database, and how your organization is using the audits from the Navigator console or API.

**How are events archived?**

Most organizations keep audit events in storage well after they are useful for casual queries through the Navigator console. Your archiving solution should balance the effort required to access archived audit data with the cost of
implementing your archiving system. Be sure that whatever system you design can handle the volume of audits you intend to maintain.

This Navigator YouTube video gives an overview of the audit tuning process and some database queries and other techniques that will help you make sure your audit system is working well.

**Reviewing Navigator Audits for Unproductive Events**

You can use the Navigator console, Audit tab to get a general feel for what events are audited and if you are seeing too many audit events that don’t add value. In addition, here are some queries you can run against the audit database itself to identify users or operations that produce more events than expected or needed.

**Review Audited Operations**

1. Log into the database as a privileged user.
   
   For example for MySQL:
   ```bash
   $ mysql -u user_name -p
   ```
   
   You are prompted for a password.

2. Show the databases and look for the Navigator database.
   ```
   mysql> show databases;
   mysql> use navigator audit server database
   ```
   
   If you aren’t sure of the name, you can check the **Navigator Audit Server Database Name** in Cloudera Manager.

3. List the tables to see that the table names show the service and the day the audits are from.
   ```
   mysql> show tables;
   ```

4. Pick a table and run a query to show the events recorded by user.
   ```
   mysql> select username, operation, count(*) as usercount from audit table group by username order by usercount DESC;
   ```
   
   If you find that a relatively large number of events were collected for a single user, it may be that this user is a system user. The default filters discard events performed by the HDFS superuser, but only if the role name is `hdfs`. You may need to add another audit filter to discard events by this additional system user name.

5. Pick a table and run a query to show the events recorded by operation.
   ```
   mysql> select operation, username, count(*) as opcount from audit table group by operation order by opcount DESC;
   ```
   
   This query may indicate that your audit system includes large numbers of events that you may not care to track. In that case, consider adding an audit filter to discard the unneeded events by operation.

**Reviewing Default Audit Filters**

To see the default Navigator Audit Filters:

1. Log in to the Cloudera Manager Admin Console.
2. Select the service you want to review. Select **Cluster > Service-Name**
3. Click the **Configuration** tab.
4. Select **Service (Service-Wide)** for the **Scope** filter.
5. Select **Cloudera Navigator** for the **Category** filter.
6. Click 🔄 for Audit Event Filter to open the property description.

The make sure the default filters are doing what they are designed to do, check that the user names in the filters correspond to the role names in your environment. For example, one default HDFS filter is configured to discard redundant audit events produced in HDFS when operations are performed in Hive or other services. This filter is configured for the user hdfs. If your environment uses other names for HDFS superusers, consider creating additional filters to discard the same operations performed under the other superuser names.

Note: Cloudera Manager version 5.16 includes new default filters for HDFS; if your HDFS service already had custom audit filter rules, these default would not be applied. If that's the case, you may want to manually apply some of these filters to better target your audit event collection. See Default and Recommended Audit Filters on page 97.

Review and Predict Audit Volume

One way to calculate the volume of disk space you’ll need for the Navigator Audit Server database is to look at the event count for a single day or average of days and multiply that volume by the number of days in the audit event expiration period. You can modify the expiration period in Cloudera Manager, in the Navigator Audit Server Data Expiration Period property. The default value is 90 days.

Calculate Audit Volume

1. To see the size of all audit tables for a given date, query against a set of tables. For example, in MySQL:

```sql
mysql> SELECT table_name AS 'Table', round(((data_length + index_length) / 1024 / 1024), 2) `Size in MB` FROM information_schema.TABLES WHERE table_schema = "navigator audit database" and table_name like "%YYYY_MM_DD%";
```

Insert your Navigator Audit Database name and a valid audit date (YYYY_MM_DD format) in the query.

2. To see the total size of audits for each day for a give year of audits, query against the event tables in the database for a given year:

```sql
SELECT sum(round(((data_length + index_length) / 1024 / 1024), 2)) `Size in MB across all audits`, right(table_name,10) as dateset FROM information_schema.TABLES WHERE table_schema = "navigator audit database" and table_name like "%YYYY_%" group by dateset;
```

Insert your Navigator Audit Database name and a valid audit year (YYYY format) in the query.

3. Calculate the expected size of your audit database by selecting a representative size for a day of audit data and multiplying that value by the number of days in the expiration period.

For example, if your system is generating about 5 GB of events per day and you are keeping the default 90 days of audits, you’ll need about 450 GB of space for the audit database.

Navigator Metadata Server Management

The Navigator Metadata Server is one of the two roles that provides Cloudera Navigator data management functionality. The Navigator Metadata Server manages, indexes, and stores entity metadata extracted from cluster services. Using policies defined by data stewards and others, the Navigator Metadata Server can tag entities with metadata or take other actions during the extraction process. It is the metadata that enables data discovery and data lineage functions for Cloudera Navigator.

This section starts with an overview of the Navigator Metadata Server system architecture and then covers administrator tasks that require Cloudera Manager Admin Console such as adding the Navigator Metadata Server role to an existing cluster, tuning for optimal performance, and setting up the LDAP or Active Directory groups to use Cloudera Navigator role-based access privileges.
Navigator Metadata Architecture

The figure below shows a high-level view of the key components that make up the Cloudera Navigator Metadata Architecture. Extracting, indexing, and storing metadata from cluster entities, both on-premises and in the cloud, are some of the important processes implemented by this architecture. As shown in the figure below, Cloudera Navigator can also extract metadata from entities stored on cloud services (Amazon S3) and from clusters running on Amazon EC2 or Cloudera Altus.

Organizations use the Cloudera Navigator console to define their own managed metadata properties to apply to entities contained in the cluster. The Navigator Metadata Server periodically connects to the Cloudera Manager Server and extracts metadata from the entities CDH services (HDFS, Hive, Impala, and so on).

Note: Not all services are shown in the image above (for example, Sqoop1).

As it extracts metadata, the Navigator Metadata Server can also apply any policies that have been configured. For example, a policy may specify that all Hive table entities with table name budget_2017 be tagged with the managed metadata property name finance-team, or that any entities with a file creation date (technical metadata) prior to 2014 be labeled with the prefix archive-yyyy, with the specific year obtained from the file creation date and applied to the label.

Extracted metadata is then indexed for later search using the embedded Solr instance. Solr Documents that comprise the index are stored on disk in the storage directory at the location configured in Cloudera Manager Admin Console for the Navigator Metadata Server datadir. See Services Supported for Metadata Extraction on page 107 for more detail.

The Navigator Metadata Server also:

- Manages authorization data for Cloudera Navigator users
- Manages audit report metadata
- Generates metadata and audit analytics
Exposes the Cloudera Navigator APIs

Hosts the web server that provides the Cloudera Navigator console. (The web server and the console are also used to support the Cloudera Navigator Auditing services.)

The Navigator Database stores policies, user authorization and audit report metadata, and analytic data. Extracted metadata and the state of extractor processes is kept in the storage directory.

Services Supported for Metadata Extraction

The Navigator Metadata Server extracts metadata for the resource types listed in the table. Metadata is "pulled" by Navigator Metadata Server or "pushed" from the service through the Cloudera Manager agent on the host where the service is running.

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Metadata Extracted</th>
<th>Extraction Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>HDFS metadata at the next scheduled extraction run after an HDFS checkpoint.</td>
<td>Pull</td>
</tr>
<tr>
<td></td>
<td>If the cluster is configured for high availability (HA), metadata is extracted at the same time it is written to the JournalNodes.</td>
<td></td>
</tr>
<tr>
<td>Hive Server 2</td>
<td>Query metadata is pushed to Navigator as queries occur. See Managing Hive and Impala Lineage Properties.</td>
<td>Push</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td>Database asset metadata (database, tables, views, partitions, columns). See Managing Hive and Impala Lineage Properties. HMS entities include tables that result from Impala queries and Sqoop jobs.</td>
<td>Pull</td>
</tr>
<tr>
<td>Impala</td>
<td>Query metadata is pushed to Navigator from the Impala Daemon lineage logs. See Managing Hive and Impala Lineage Properties.</td>
<td>Push</td>
</tr>
<tr>
<td>MapReduce</td>
<td>Job metadata from the JobTracker. The default setting in Cloudera Manager retains a maximum of five jobs; if you run more than five jobs between Navigator extractions, the Navigator Metadata Server extracts the five most recent jobs.</td>
<td>Pull</td>
</tr>
<tr>
<td>Oozie</td>
<td>Oozie workflows from the Oozie Server.</td>
<td>Pull</td>
</tr>
<tr>
<td>Pig</td>
<td>Pig script runs from the JobTracker or Job History Server.</td>
<td>Pull</td>
</tr>
<tr>
<td>S3</td>
<td>Bucket and object metadata.</td>
<td>Pull</td>
</tr>
<tr>
<td>Spark</td>
<td>Query metadata is pushed to Navigator as queries occur. Spark job metadata from YARN logs is pulled as part of the YARN extractor. Note that Spark 1.6, which is included with CDH, and CDS 2 Powered by Apache Spark include integration with Navigator. Third-party versions of Spark do not produce metadata for Navigator lineage.</td>
<td>Push</td>
</tr>
<tr>
<td>Sqoop 1</td>
<td>Query metadata is pushed to Navigator as queries occur. Database and table metadata comes from the HMS extraction.</td>
<td>Push</td>
</tr>
<tr>
<td>YARN</td>
<td>Job metadata from the ResourceManager.</td>
<td>Pull</td>
</tr>
</tbody>
</table>

Metadata Extraction Timing

The two types of Navigator extractors have different timing: Push extractors send metadata to Navigator after each event that generates metadata (Hive Server 2, Impala, Spark, Sqoop). Pull extractors run about every 10 minutes, when Navigator Metadata Server triggers them (Oozie, YARN, HDFS, HMS, S3). When the 10 minute mark comes around for a given service, Navigator checks to see if an extraction process is already running; if one is, Navigator resets the wait time without starting another extraction. Extractors for different services typically run simultaneously, inserting new metadata into Navigator in a regular cadence. However, extractor behavior and timing for HMS and HDFS can be more complicated because of the volume of metadata they collect in a single extractor run.
HMS Extraction

Navigator extracts metadata from Hive Metastore (HMS) through a pull extractor where Navigator Metadata Server triggers the extraction. Every time Navigator extracts HMS metadata, it reviews all databases for all data assets: tables, views, partitions, etc. There is no incremental extraction from HMS. When your HMS includes many databases and lots of assets, the extraction can take a long time: certainly more than the 10 minute extraction interval.

As it progresses through the extraction, Navigator creates new entities for any data assets that aren’t already represented in Navigator.

HDFS Extraction

Navigator extracts metadata for HDFS from the namenode. After the initial extraction, Navigator users iNotify to extract changes to HDFS contents rather than re-extracting all entities again. The initial HDFS extraction process still requires a long period; subsequent extractions may be more likely to conform to the 10-minute pattern. If Navigator encounters an iNotify failure, it will fall back to bulk extraction.

Calculating Lineage from Metadata

As described in detail in Lineage Generation Architecture on page 67, lineage is generated when Navigator combines the metadata for data assets (such as file, tables, views) with the metadata for operation executions (queries and jobs). Tables and views come from HMS extraction, while operation executions come from push extractors for Impala, Hive Server 2, Spark, YARN, and Sqoop. If any of the metadata is missing (not extracted yet), lineage relationships aren’t calculated or displayed. For lineage relationships to be displayed completely, Navigator must have completed extraction from each of the services involved and then it must run a lineage calculation process called "linking."

To understand the timing of how metadata and lineage relationships become available in the Navigator console, consider the pull extractor for HMS and a push extractor from Hive Server 2 as they run independently. Lineage metadata, which requires metadata from both HMS and a query engine, appears in the Navigator console only after both extraction operations are complete and the linker process runs. The following illustration describes the timing for each of the processes that must complete for lineage to appear in the console:

1. HMS Extraction is a long-running process that steps through HMS one database at a time.
2. Query engines such as Hive Server 2 push metadata updates to Navigator as soon as an event occurs: In this example, a query that creates table X is pushed to Navigator and Navigator creates an operation entity and an operation execution entity to represent the query. Navigator creates lineage relationship entities for the new table, but the relationships are left "unlinked" because the new table doesn’t yet exist in Navigator.
3. The HMS extraction gets to the database where the table was created and passes the metadata to Navigator, which creates a new entity for the table.
4. After the HMS extraction finishes, Navigator triggers a linker process that reviews the "unlinked" lineage relations and connects them to the correct entities. Now the table created in step 2 is included in the lineage relationship created to describe the query.

5. Lineage for the new table appears in the Navigator console.

Note that step 3 is a variable here: it’s possible that the HMS extraction run may have already processed the relevant database (1) before the query occurred (2). In that case, the HMS extraction would complete and linking would run (3), but there would not be an entity in place for the new table. Lineage for the new table doesn’t appear until the next HMS extraction run reaches the relevant database (4) to create the table entity; then linking runs at the end of the HMS extraction (5) and adds the correct lineage information (6).

Troubleshooting

If Navigator is missing entities for:

- Queries: check the push extractor processing. The Navigator logs won’t show any problems in this case. Instead, check the Cloudera Manager agent log on the host where the suspect service is running.
- Data asset entities: check the HMS pull extractor processing in the Navigator Metadata Server log.
- Filesystem entities: check the HDFS or S3 extractor processing in the Navigator Metadata Server log.

If Navigator is missing lineage, make sure that Navigator shows the entities for both the data assets and the queries that describe the lineage; then give enough time for the lineage processing to complete. If you still don’t see lineage relations, check to make sure the linker is running (in the Navigator Metadata Server log). If HMS extraction never completes, you may see metadata for some data assets, but the linking process may not have been triggered. Check the Navigator Metadata Server log.

The pull extraction frequency can be configured using nav.extractor.poll_period in the Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties in Cloudera Manager. Note that lowering the polling period won’t affect latency if your system has one of the problems described here.

To find pull extractor events in the Navigator Metadata Server log, look for "extraction tasks for service”, then filter further on the specific services you are interested in. For example:

```
[[ExtractorServiceExecutor-3] Cluster 1::HIVE-1]: Enqueuing 6 extraction tasks for service 'Cluster 1::HIVE-1'...
[[ExtractorServiceExecutor-3] Cluster 1::HIVE-1]: All cdh-client extraction tasks for service 'Cluster 1::HIVE-1' finished.
[[ExtractorServiceExecutor-7] Cluster 1::HIVE-1]: All cdh-client extraction tasks for service 'Cluster 1::HIVE-1' finished.
[[ExtractorServiceExecutor-7] Cluster 1::HIVE-1]: All cdh-client extraction tasks for service 'Cluster 1::HIVE-1' finished.
```
In this example, it took 10 minutes to finish the first extraction (20:26:00 - 20:15:58). Navigator waited 10 minutes to start the second extraction (20:35:58). The second extraction took 20 minutes to finish (20:56:00 - 20:35:58). The third extraction started 10 minutes after the second one finished (21:05:59), a total of 30 minutes after the start time of the second extraction.

Metadata Indexing

After metadata is extracted, the embedded Solr engine indexes it and makes it available for searching. The Solr instance indexes entity properties and the relations among entities. Relationship metadata is implicitly visible in lineage diagrams and explicitly available by downloading the lineage using the Cloudera Navigator APIs.

Data stewards and other business users explore entities of interest using the Cloudera Navigator console. Metadata and lineage that has been extracted, linked, and indexed can be found using Search. See Lineage, Metadata and other sections in this guide for Cloudera Navigator console usage information. The following sections focus on using the Cloudera Manager Admin Console for systems management tasks.

Setting Up Navigator Metadata Server

In addition to the basic configuration topics discussed below, see the Navigator Metadata Server Tuning for memory sizing and other considerations.

Setting up Navigator Role Instances on Different Hosts

By default, the Cloudera Manager installation wizard assigns the Navigator Audit Server and Navigator Metadata Server roles to the same Cloudera Management Service host. This configuration can support small clusters but larger production clusters often need these roles distributed to different nodes for better performance. You can distribute these roles to different nodes at install time or you can move the Navigator Metadata Server later if necessary.

Adding the Navigator Metadata Server Role

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

1. Configure the database where policies, roles, and audit report metadata is stored. For production environments, Cloudera supports only external databases. The internal PostgreSQL database should be used for non-production clusters only, such as proof-of-concept.
   - For external databases (production environments): The database should be up and running before you add the Navigator Metadata Server role to the cluster. You must have the database schema name (user name), password, and other details to provide when adding the role. When you have these ready, you can proceed.
   - For internal databases (non-production environments): No additional preliminary information is needed. You can proceed.

2. Log in to the Cloudera Manager Admin Console.
4. Click the Instances tab.
5. Click the Add Role Instances button. The Customize Role Assignments page displays and lists the nodes available to support additional roles, assuming the cluster has available resources. Reassign and customize role instances as needed.
6. Select the Navigator Metadata Server role and assign it to appropriate host for your cluster.
7. When finished, click Continue. The Database Setup page displays.
8. Click Use Custom Database.
9. In the Navigator Metadata Server section, enter the details of your database instance:
   - Database host name
   - Database type
   - Database name
• Username
• Password

Note: Do not use the embedded database for production deployments. It is not a supported configuration. For non-production environments only you can select Use Embedded Database to have Cloudera Manager create and configure the database for you. Make a note of the generated passwords.

Here is an example of a configured custom (external) database instance that uses MySQL for the Navigator Metadata Server database:

10. Click Test Connection to verify the communication between the cluster and the external database. If the test fails, check the database settings and try again. If you selected embedded database, a message displays regarding database creation.

11. Click Continue.


13. Click Finish.

Removing the Navigator Metadata Server Role

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

1. Log in to the Cloudera Manager Admin Console.

2. Select Clusters > Cloudera Management Service.

3. Click the Instances tab.

4. Select the Navigator Metadata Server role type.

5. If the Navigator Metadata Server role is not already stopped, choose Actions for Selected > Stop.

Click Stop to confirm the action.

6. Select the Navigator Metadata Server role type.

7. Choose Actions for Selected > Delete.
Starting, Stopping, and Restarting a Navigator Metadata Server Role Instance

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Instances tab.
4. Click the link for the Navigator Metadata Server from the Role Type list. The Actions for Selected button activates and displays (1) for the selected role.
5. From the Actions for Selected (1) menu, select the Action you want to perform on this role instance:
   - Start
   - Stop
   - Restart
   - Enter Maintenance Mode
   - Exit Maintenance Mode
   - Delete

A confirmation message displays giving you the option to Cancel or complete the specified Action.

6. Click Cancel to abandon the process, or click Action to execute the action.

Backup Navigator Metadata Server Data

Navigator Metadata Server stores metadata in an embedded Solr data directory and in a database instance. Back up these locations together so the backups describe the same metadata state.

1. Shut down the Navigator Metadata Server role.
2. Backup the data directory.
   - The Navigator Metadata Server default directory is /var/lib/cloudera-scm-navigator. To confirm that this is the path your deployment uses, log in to the Cloudera Manager Admin Console and search for the Navigator Metadata Server Storage Dir property (nav.data.dir).
3. Backup the database instance.
   - Confirm the Navigator Metadata Server database location and connection information.
     To locate this information, log in to the Cloudera Manager Admin Console and go to Clusters > Cloudera Management Service. Click the Configuration tab. Select Scope > Navigator Metadata Server. Select Category > Database. The results indicate the type, host, database name, and login account.
     - Using the appropriate tool for the database instance (such as MySQL), backup the database. See Backing up Cloudera Manager.
4. Start the Navigator Metadata Server role.

Moving the Navigator Metadata Server to a New Host

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

When resources become tight on the host where Navigator servers are installed, one solution is to move one or both servers to another host or hosts. These instructions describe moving only the Navigator Metadata Server. Be sure to stop the Navigator Metadata Server before backing up the storage directory.

1. Stop the Navigator Metadata Server role.
2. Backup the Navigator Metadata Server database and storage directory.
3. Delete the Navigator Metadata Server role from the existing host.
4. If the storage directory is not on NFS/SAN, move the data to the same path on the new host.
By default, the Navigator Metadata Server storage directory is /var/lib/cloudera-scm-navigator. This value is set in the Cloudera Manager Navigator Metadata Server Storage Dir property (nav.data.dir). If you want to put the directory in a different location, change the property when you reconfigure the Navigator Metadata Server role and before you restart the role.

5. Add the role to a new host.

This step includes configuring the new storage directory location and specifying the database connection information. See Adding the Navigator Metadata Server Role on page 110 for details.

6. Make sure the Navigator Metadata Server storage directory is configured correctly for the new host. See Configuring the Navigator Metadata Server Storage Directory on page 114.

**Important:** Make sure that the storage directory is in place and the Cloudera Manager configuration is correct before starting the Navigator Metadata Server role on the new host. If Navigator Metadata Server starts without finding the storage directory, it will create a new one, which will not be synchronized with the database and the move will fail. If this happens, stop the server, replace the newly created storage directory with the backup from step 2 and restart the server.

7. Start the Navigator Metadata Server role.

Setting the Navigator Metadata Server Java Heap Size

**Cloudera Manager Required Role:** Navigator Administrator (or Full Administrator)

Use the formula discussed in Memory Sizing Considerations for Navigator Metadata Server to calculate optimal Java heap size:

\[
\text{((num_nav_elements + num_nav_relations) * 200 bytes) + 2 GB}
\]

Briefly, the total of num_nav_elements and num_nav_relations is the number of Solr documents (per the Navigator Metadata Server log file) that comprise the index. See sizing guidelines for complete details. If you are just starting your Navigator deployment, set the Java heap size to at least 10 GB and as much as 30 GB if you have the memory available on the Navigator host.

After you have determined an appropriate Java heap setting using this or another technique, you can change the setting as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. For Filter Category, click Resource Management to display the Java heap size property. The default value is 1.5 GiB (specified in binary units).

<table>
<thead>
<tr>
<th>Java Heap Size of Navigator Metadata Server in Bytes</th>
<th>Navigator Metadata Server Default Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1536 MiB</td>
<td>1.5 GiB</td>
</tr>
</tbody>
</table>

6. Use the drop-down selector to change the unit scale to B (Bytes), KiB, MiB, or GiB and set your preferred heap size.

7. Click Save Changes. The setting takes effect only after restarting the role. Restart the Navigator Metadata Server role now or make other configuration changes and restart after you are finished with all changes. To restart the role:

8. Click the Instances tab to display the list of instances of each role type.

9. Click the link for the Navigator Metadata Server to select it.
From the Actions for Selected (1) menu, select Restart to restart the Navigator Metadata Server role instance using the new Java heap size.

Configuring the Navigator Metadata Server Storage Directory

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

Navigator Metadata Server uses a Solr instance (a search engine) to index extracted metadata for fast search and retrieval. The searchable Solr documents are stored in a directory on the host on which the Navigator Metadata Server runs. The default location for the directory is /var/lib/cloudera-scm-navigator.

To change the location of the storage directory:

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Instances tab.
4. Click the link for the Navigator Metadata Server to select it.
5. From the Actions for Selected (1) menu, select Stop.
6. Open a terminal session and log in to the host configured to run Navigator Metadata Server.
7. Copy the contents of the current storage directory to the new location. When copying is complete, return to the Cloudera Manager Admin Console.
8. Click the Configuration tab.
10. Enter the directory path in the Navigator Metadata Server Storage Dir property. The property applies to the default role group, but you can apply to other role groups as needed. See Modifying Configuration Properties Using Cloudera Manager for details.
11. Click Save Changes. This change does not take effect until the Navigator Metadata Server role is restarted.
12. Click the Instances tab.
13. Click the link for the Navigator Metadata Server from the list of instances.
14. From the Actions for Selected menu, select Restart to restart the Navigator Metadata Server role instance.

Configuring the Navigator Metadata Server Port

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

The Cloudera Navigator console is accessed at port 7187 by default, on the host supporting the Navigator Metadata Server role. Follow the steps below to change setting.

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. In the Search box, type the word "ports" to display properties.
5. Specify the port in the Navigator Metadata Server Port property. The property applies to the default role group, but you can apply to other role groups as needed. See Modifying Configuration Properties Using Cloudera Manager for details.
6. Click Save Changes.
7. Click the Instances tab.
8. Restart the Navigator Metadata Server role.

Configuring the Navigator Metadata Server Log Directory

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

The default location of the Navigator Metadata Server logs is /var/log/cloudera-scm-navigator, but you can change to a different location, as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
4. Under the Filter for **Scope**, click **Navigator Metadata Server**.

5. Under Filter **Category**, click **Logs**.

6. Set the **Navigator Metadata Server Log Directory** property. The property applies to the default role group, but you can apply to other role groups as needed. See [Modifying Configuration Properties Using Cloudera Manager](#) for details.

7. Click **Save Changes**.

8. Restart the Navigator Metadata Server role instance.

**Configuring the Navigator Metadata Server Session Timeout**

**Cloudera Manager Required Role:** **Navigator Administrator** (or **Full Administrator**)

The default session timeout is 30 minutes (1800 seconds). To change the timeout period:

1. Log in to the Cloudera Manager Admin Console.

2. Select **Clusters > Cloudera Management Service**.

3. Click the **Configuration** tab.

4. Under the Filter for **Scope**, click **Navigator Metadata Server**.

5. For Filter **Category**, click **Advanced**. Scroll to find the Advanced Configuration Snippet (Safety Valve) entry field for `cloudera-navigator.properties`.

6. Set the value for the `nav.max_inactive_interval` property to the number of seconds that can elapse before system times out:

   ```
   nav.max_inactive_interval=num-seconds-until-timeout
   ```

   This example reduces the timeout period from the default of 30 minutes to 15 minutes:

7. Click **Save Changes**.

8. Restart the Navigator Metadata Server role instance.

**Disabling Anonymous Usage Collection**

**Cloudera Manager Required Role:** **Navigator Administrator** (or **Full Administrator**)

Cloudera Navigator by default sends anonymous usage information to Cloudera for analysis. The usage information helps Cloudera improve Cloudera Navigator functionality and pro-actively identify areas of the product to enhance. The information is completely anonymous and cannot be traced to any customer installation.

To disable anonymous usage information collection:

1. Log in to the Cloudera Manager Admin Console.

2. Select **Clusters > Cloudera Management Service**.

3. Click the **Configuration** tab.

4. Under the Filter for **Scope**, click **Navigator Metadata Server**.

5. Click the **Allow Usage Data Collection** to clear the checkbox.

6. Click **Save Changes**.

7. Restart the Navigator Metadata Server role instance.

Cloudera Navigator stops emitting anonymous usage information.

This capability can be re-enabled at any time by clicking the Allow Usage Data Collection box that has been cleared.
Configuring Navigator Metadata Server for Kerberos and Custom Keytab Retrieval

If Navigator is running on a Kerberized cluster and the cluster is set up to use a custom keytab retrieval script, you need to manually put keytabs for HDFS and Hue principals into the Navigator home directory (typically /var/lib/cloudera-scm-navigator) for use by the Navigator web server.

If the keytabs are not available, you'll see Navigator Metadata Server log errors such as the following:

```
java.lang.RuntimeException: java.util.concurrent.ExecutionException: java.lang.RuntimeException: java.io.IOException: Login failure for hdfs/node.example.com@EXAMPLE.COM from keytab /var/lib/cloudera-scm-navigator/temp/1234/hadoop1234.keytab:
javax.security.auth.login.LoginException: Unable to obtain password from user
```

Navigator Metadata Server Configuration Settings Summary

Access to the Cloudera Manager Admin Console and the ability to change any of the settings in the table below requires the Cloudera Manager user role of Navigator Administrator or Full Administrator.

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Navigator Metadata Storage Dir</td>
<td>/var/lib/cloudera-scm-navigator</td>
<td>Change to another path if you like. Be sure to move existing content from the default directory to the new directory to maintain state. Offload content from this directory prior to upgrading for faster upgrades.</td>
</tr>
<tr>
<td>Cloudera Navigator Server Port</td>
<td>7187</td>
<td>Change to another port if necessary. This is the port used to access Cloudera Navigator console.</td>
</tr>
<tr>
<td>Navigator Metadata Server Log Directory</td>
<td>/var/log/cloudera-scm-navigator</td>
<td>Examine the log to find out how many documents the Solr index contains, and use that count to estimate Java heap size needed for normal operations and for upgrades. Use the log to troubleshoot issues.</td>
</tr>
<tr>
<td>Server Session Timeout</td>
<td>30 minutes (1800 seconds)</td>
<td>Timeout is specified in number of seconds that must elapse before the system generates a timeout event.</td>
</tr>
<tr>
<td>Allow Usage Data Collection</td>
<td>enabled</td>
<td>Cloudera Navigator uses Google Analytics to anonymous usage data to Cloudera for product enhancement. Disable this setting if you do not want to send usage data.</td>
</tr>
</tbody>
</table>

Navigator Metadata Server Tuning

This page can help you tune your Navigator Metadata Server instance for peak performance. See also Setting up Navigator Role Instances on Different Hosts.

Memory Sizing Considerations for Navigator Metadata Server

Unlike Navigator Audit Server for which configured Java heap size is rarely an issue, Navigator Metadata Server encompasses two processes that have a direct impact on memory, specifically:

- Extracting metadata from the cluster and creating relationships among the metadata entities (facilitating lineage)
- Querying to find entities

Navigator Metadata Server uses Solr to index, store, and query metadata. Indexing occurs during the extraction process, with the resulting Solr documents—data structure used by Solr for index and search—stored in the specified Navigator
**Metadata Server Storage Dir.** Because the metadata is indexed, querying is fast and efficient. However, Solr indexing runs in-process with Navigator Metadata Server, so the amount of memory configured for the Java heap is critical.

There is a direct correlation between the number of Solr documents contained in the index and the size of the Java heap required by Navigator Metadata Server, so this setting may need to be changed over time as the number of Solr documents making up the index increases. To calculate optimal Java heap setting for your system, see [Estimating Optimal Java Heap Size Using Solr Document Counts](#).

**Estimating Optimal Java Heap Size Using Solr Document Counts**

Each time it starts up, Navigator Metadata Server counts and logs the number of Solr documents in the datadir (the Navigator Metadata Server storage directory), as shown below:

```plaintext
```

These counts can be used to estimate optimal Java heap size for the server, as detailed below. If your normal setup provides less than 8 GB for the Navigator Metadata Server heap, consider increasing the heap before performing an upgrade. See [Setting the Navigator Metadata Server Java Heap Size](#) for details about using Cloudera Manager Admin Console to modify this setting when needed.

**Note:** If your system has been running without interruption for awhile, the data in the log may be stale. Restart the Navigator Metadata Server when it is convenient and then look at the log to find the current number of documents to use in the calculation.

**To estimate the Java heap size:**

1. Open the log file for Navigator Metadata Server. By default, logs are located in `/var/log/cloudera-scm-navigator`.
2. Find the number of documents in `solr core nav_elements` line in the log.
3. Find the number of documents in `solr core nav_relations` line in the log.

**Warning:** If the number of relations is more than 2 times the number of elements, you may have encountered the problem described by [TSB-259 Upgrade needed for Cloudera Navigator due to high risk of failure through exhausting data directory resources](#). Follow the instructions for mitigating the problem rather than simply changing your Java heap.

4. Multiply the total number of documents by 200 bytes per document and add to a baseline of 2 GB:

   \[(\text{num\_nav\_elements}\times200\ \text{bytes}) + 2\ \text{GB}\]

   For example, using the log shown above, the recommended Java heap size is ~14 GiB:

   \[(68813088\times200) + 2\ \text{GB} = 13762617600\ \text{bytes} = \sim12.8\ \text{GiB} + 2\ \text{GB} = \sim14-15\ \text{GiB}\]
**Important:** If you are upgrading to a Cloudera Manager version 5.10.0 or later, you’ll need considerably more Java heap to perform the upgrade. Upgrading to the 5.10 release includes the unusual performance of a schema change in Solr. This operation requires additional Java heap for the duration of the upgrade process. To calculate the recommended Java heap for the upgrade, multiply the total number of documents by 200 bytes per document and add to a baseline of 2 GB:

\[
((\text{num\_nav\_elements} + \text{num\_nav\_relations}) \times 200 \text{ bytes}) + 2 \text{ GB}
\]

For example, using the log shown above, the recommended Java heap size is ~31 GiB:

\[
((68813088 + 78813930) \times 200) + 2 \text{ GB} = 29525403600 \text{ bytes} = \sim 29.5 \text{ GiB} + 2 \text{ GB} = \sim 31.3 \text{ GiB}
\]

Reducing the Metadata collected by Navigator Metadata Server

You can choose to leave some file system paths out of the scope of information tracked in Cloudera Navigator. Cloudera Manager provides a blacklist where you can specify file systems paths that should be filtered out of metadata extracted from HDFS and S3.

**To filter file system paths from tracked metadata:**

1. Log in to Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. Select Extractor Filter for the Category filter.
6. Enable the filter:
   - HDFS Filter Enable
   - S3 Filter Enable
7. In the appropriate filter list, include the file system path that you want to exclude from Navigator Metadata Server tracking:
   - HDFS Filter Blacklist
   - S3 Filter list
   
   The entry can be a specific path or a Java regular expression specifying a path. For example, to specify a directory and all subdirectories, use an expression such as

   /path/to/dir/(?:/.*)?

8. Enter additional entries in the filter list by clicking + to open another entry.
9. For S3, set the S3 Filter Default Action to DISCARD.
10. Click Save Changes.
11. Click the Instances tab.
12. Restart the role.

Purging the Navigator Metadata Server of Deleted and Stale Metadata

Administrators can manage the Java heap requirements by clearing the Navigator Metadata Server of stale and deleted metadata prior to an upgrade or whenever system performance seems slow. Purging stale and deleted metadata also helps speed up display of lineage diagrams. Purge fully removes metadata that has been deleted.

For Cloudera Navigator 2.11 (and higher) releases—Cloudera Navigator console (Administration tab) provides a fully configurable Purge Settings page. See Managing Metadata Storage with Purge for details.
For Cloudera Navigator 2.10 (and prior releases)—The Purge capability can be directly invoked using the Cloudera Navigator APIs. See Using the Purge APIs for Metadata Maintenance Tasks for details.

### Hive and Impala Lineage Configuration

**Cloudera Manager Required Role:** [Configurator](#) (or [Cluster Administrator](#), or [Full Administrator](#))

Unlike for other services running in the cluster (such as Pig), lineage data from Hive and Impala queries is not extracted by Navigator Metadata Server. Instead, these two services write query data to log files collected in a specific directory on the cluster node. The Cloudera Manager Agent process running on that node monitors the directory and routinely sends the log files to the Navigator Metadata Server, where the query data is coalesced with other metadata collected by the system.

Lineage collection from Hive and from Impala log files is enabled by default—each of these services has its own **Enable Lineage Collection** property and some related configuration properties, which can be disabled or reconfigured as detailed below.

#### Modifying Lineage Collection Settings for Hive

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Lineage Collection</td>
<td>Enabled for Hive, Service-Wide</td>
<td>Enable collection of lineage from the service’s roles.</td>
</tr>
<tr>
<td>Hive Lineage Log Directory (lineage_event_log_dir)</td>
<td>/var/log/hive/lineage</td>
<td>Directory in which Hive lineage log files are written.</td>
</tr>
<tr>
<td>Hive Maximum Lineage Log File Size (max_lineage_log_file_size)</td>
<td>100 MiB</td>
<td>Maximum size (MiB, GiB) of Hive lineage log file before a new file is created.</td>
</tr>
</tbody>
</table>

To disable Hive lineage collection:

control whether the Impala Daemon role logs to the lineage log and whether the Cloudera Manager Agent collects the Hive and Impala lineage entries:

1. Log in to Cloudera Manager Admin Console.
2. Select **Clusters > Hive**.
3. Click the **Configuration** tab.
4. Type **lineage** in the Search box.
5. Click the **Enable Lineage Collection** check-box to deselect it and disable lineage collection.
6. Click **Save Changes**.
7. Restart the Hive service.

#### Modifying Lineage Collection Settings for Impala

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Impala Lineage Generation (enable_lineage_log)</td>
<td>Enabled for the Impala daemon default group</td>
<td>When enabled, Impala daemon process creates a logfile containing lineage data and stores it in the directory specified by the <strong>Impala Daemon Log Lineage Directory</strong> property.</td>
</tr>
<tr>
<td>Enable Lineage Collection</td>
<td>Enabled for Impala Service-Wide</td>
<td>Enable collection of lineage from the service’s roles.</td>
</tr>
<tr>
<td>Impala Daemon Lineage Log Directory (lineage_event_log_dir)</td>
<td>/var/log/impalad/lineage</td>
<td>Directory in which Impala daemon lineage log files are written. When <strong>Impala Lineage Generation</strong> property is enabled, Impala generates its lineage logs in this directory.</td>
</tr>
</tbody>
</table>
### Maximum Lineage Log File Size (max_lineage_log_file_size)

The `max_lineage_log_file_size` property determines whether lineage logs should be collected by the Cloudera Manager Agent. To control whether the Impala Daemon role logs to the lineage log and whether the Cloudera Manager Agent collects the Hive and Impala lineage entries:

To disable lineage collection for Impala queries:

1. Log in to Cloudera Manager Admin Console.
2. Select Clusters > Impala.
3. Click the Configuration tab.
4. Type `lineage` in the Search box.
5. Click the Enable Lineage Collection check-box to deselect it and disable lineage collection.
6. Click the Enable Impala Lineage Generation check-box to deselect it.
7. Click Save Changes.
8. Restart the Impala service.

De-selecting either Enable Lineage Collection or Enable Impala Lineage Generation disables lineage collection for Impala.

### Configuring Hive on Spark and Impala Daemon Lineage Logs

If the value of a log directory property is changed, and service is restarted, the Cloudera Manager Agent starts monitoring the new log directory. In this case it is possible that not all events are published from the old directory. To avoid losing lineage information when this property is changed, perform the following steps:

1. Stop the affected service.
2. Copy the lineage log files and (for Impala only) the `impalad_lineage_wal` file from the old log directory to the new log directory. This needs to be done on the HiveServer2 host and all the hosts where Impala Daemon roles are running.
3. Start the service.

To edit lineage log properties:

1. Go to the service.
2. Click the Configuration tab.
3. Type `lineage` in the Search box.
4. Edit the lineage log properties.
5. Click Save Changes.
6. Restart the service.

### Configuring and Managing Extraction

Entities extracted from cluster services and the metadata that is applied to them supports Cloudera Navigator features, such as the ability to trace entities to their source in lineage diagrams, search for specific entities, and so on. Extraction is enabled for some services by default, while for other services, extraction must be specifically enabled. For example, extraction is enabled for Spark by default. Extraction consumes computing resources, such as memory and storage, so administrators may want to disable extraction for some services entirely, or configure more selective extraction for specific services.

In addition to configuring extraction for specific services, specific filters can be configured to blacklist specific HDFS paths to remove them from the extraction process which both speeds up the process and cuts down on indexing time, as well as the amount of storage consumed by the datadir. Filters can also be configured to blacklist or whitelist Amazon S3 buckets.
Note: To ensure that the extraction process runs optimally, data stewards should regularly delete metadata for entities no longer needed. Deleting metadata should be followed by using the purge function, which cleans the datadir of Solr documents associated with the deleted entities. Using purge on a regular basis facilitates fast search and noise-free lineage diagrams.

Cloudera Manager Required Role: Navigator Administrator (or Full Administrator)

Enabling Hive Metadata Extraction in a Secure Cluster

To extract entities from Hive, the Navigator Metadata Server authenticates to the Hive metastore using the hue user account. By default the hue user has privileges and can log in to the Hive metastore for authentication. That means that if the Hive metastore configuration was changed from the defaults, user hue may not be able to authenticate and the Navigator Metadata Server will be prevented from extracting from Hive. The specific configuration properties are the following two, which work together for authentication for Hive:

- Hive Metastore Access Control and Proxy User Groups Override (Inherits from Hive Proxy User Groups when left empty (the default)
- Hive Proxy User Groups

1. Log in to Cloudera Manager Admin Console.
2. Select Clusters > Hive-1.
3. Click the Configuration tab.
4. Select Proxy for the Category filter.
5. Add hue to the Hive Metastore Access Control and Proxy User Groups Override list if necessary:
   - Click the plus icon to open an entry field to add a row to the property.
   - Type hue in the entry field.

6. Click Save Changes, and repeat the process to add the Hive Proxy User Groups property on the HDFS server:
   - Select Clusters > Hive-1.
   - Click the Configuration tab.
   - Select Proxy for the Category filter.
   - Add hue to the Hive Proxy User Groups list by adding a row and typing hue in the entry field.

7. Click Save Changes.
8. Restart the Hive service.

Disabling Spark Metadata Extraction

Metadata and lineage for Spark is extracted by default (see Apache Spark Known Issues for limitations). To disable Spark metadata extraction:

1. Log in to Cloudera Manager Admin Console.
2. Select Clusters > Spark_on_YARN-1.
3. Click the Configuration tab.
4. Select Cloudera Navigator for the Category filter. The state of the Enable Lineage Collection feature displays, as in this example:

   ![Enable Lineage Collection](image)

5. To disable lineage collection, click the checked box.
Removing Manually Enabled Lineage Collection Property

Prior to Cloudera Navigator 2.10 (Cloudera Manager 5.11), enabling lineage collection from Spark required setting a safety valve. If the cluster was upgraded from a previous release of Cloudera Navigator and an Advanced Configuration Snippet (Safety Valve) was used to enable lineage, you must remove that snippet to avoid conflict with the new Enable Lineage Collection property. Using a safety valve to enable Spark metadata extraction has been deprecated.

**Note:** Applies only to systems with safety valve configured to enable lineage collection from Spark prior to the default configuration provided in Cloudera Navigator 2.10.

To remove the deprecated safety valve setting:

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. Select Advanced for the Category filter.
6. Scroll to the Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties setting.
7. Remove any deprecated setting for Spark extraction, such as:

   ```
   nav.spark.extraction.enable
   ```
8. Click Save Changes.
9. Restart the Navigator Metadata Server role.

Filtering File System Metadata

You can choose to leave some file system paths out of the scope of information tracked in Cloudera Navigator. Cloudera Manager provides a blacklist where you can specify file systems paths that should be filtered out of metadata extracted from HDFS and S3.

To filter file system paths from tracked metadata:

1. Log in to Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. Select Extractor Filter for the Category filter.
6. Enable the filter:
   - HDFS Filter Enable
   - S3 Filter Enable
7. In the appropriate filter list, include the file system path that you want to exclude from Navigator Metadata Server tracking:
   - HDFS Filter Blacklist
   - S3 Filter list
   
   The entry can be a specific path or a Java regular expression specifying a path. For example, to specify a directory and all subdirectories, use an expression such as

   ```
   /path/to/dir(:/.*)?
   ```
8. Enter additional entries in the filter list by clicking to open another entry.
9. For S3, set the S3 Filter Default Action to DISCARD.
10. Click Save Changes.
11. Click the **Instances** tab.
12. Restart the role.

**Editing MapReduce Custom Metadata**

You can associate custom metadata with arbitrary configuration parameters to MapReduce jobs and job executions. The configuration parameters to be extracted by Cloudera Navigator can be specified statically or dynamically.

To specify configuration parameters statically for all MapReduce jobs and job executions, do the following:

1. Log in to Cloudera Manager Admin Console.
2. Select **Clusters > Cloudera Management Service**.
3. Click the **Configuration** tab.
4. Select **Navigator Metadata Server** for the **Scope** filter.
5. Select **Advanced** for the **Category** filter.
6. Scroll to find the **Navigator Metadata Server Advanced Configuration Snippet for cloudera-navigator.properties** and enter the custom properties.
7. Specify values for the following properties:
   - `nav.user_defined_properties` - A comma-separated list of user-defined property names.
   - `nav.tags` - A comma-separated list of property names that serve as tags. The property `nav.tags` can point to multiple property names that serve as tags, but each of those property names can only specify a *single* tag.
8. Click **Save Changes**.
9. Click the **Instances** tab.
10. Restart the role.
11. In the MapReduce job configuration, set the value of the property names you specified in step 7.

To specify configuration parameters dynamically:

1. Specify one or more of the following properties in a job configuration:
   - *Job properties (type:OPERATION)*
     - `nav.job.user_defined_properties` - A comma-separated list of user-defined property names
     - `nav.job.tags` - A comma-separated list of property names that serve as tags
   - *Job execution properties (type:OPERATION_EXECUTION)*
     - `nav.jobexec.user_defined_properties` - A comma-separated list of user-defined property names
     - `nav.jobexec.tags` - A comma-separated list of property names that serve as tags

   The properties `nav.job.tags` and `nav.jobexec.tags` can point to multiple property names that serve as tags, but each of those property names can only specify a *single* tag.
2. In the MapReduce job configuration, set the value of the property names you specified in step 1.

**Setting Properties Dynamically**

Add the tags `onetag` and `twotag` to a job:

1. Dynamically add the `job_tag1` and `job_tag2` properties:
   ```java
   conf.set("nav.job.tags", "job_tag1, job_tag2");
   ```

2. Set the `job_tag1` property to `onetag`:
   ```java
   conf.set("job_tag1", "onetag");
   ```
Set the `job_tag2` property to `twotag`:

```java
conf.set("job_tag2", "twotag");
```

Add the tag `atag` to a job execution:

1. Dynamically add the `job_tag` property:

```java
conf.set("nav.jobexec.tags","job_exec_tag");
```

2. Set the `job_exec_tag` property to `atag`:

```java
conf.set("job_exec_tag", "atag");
```

Add the user-defined key `key` with the value `value`:

1. Dynamically add the user-defined key `key`:

```java
conf.set("nav.job.user_defined_properties", "key");
```

2. Set the value of the user-defined key `key` to `value`:

```java
conf.set("key", "value")
```

Enabling Inputs and Outputs to Display

The Cloudera Navigator console displays a Details page for selected entities. Details include an entity's type and can optionally include table inputs and operation inputs and outputs. The inputs and outputs are not displayed by default because rendering them can slow down the display. Enabling the display of inputs and outputs in the Details page requires changing the `nav.ui.details_io_enabled` on the Navigator Metadata Server to true, as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select Clusters > Cloudera Management Service.
3. Click the Configuration tab.
5. Select Advanced for the Category filter.
6. In the Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for `cloudera-navigator.properties`, enter the following:

```java
nav.ui.details_io_enabled=true
```

7. Click Save Changes.
8. Restart the Navigator Metadata Server role.

Configuring the Server for Policy Messages

Cloudera Navigator can send notifications to a JMS message queue whenever entities change as a result of policies set for that class of entity. To use this capability, an administrator must setup a JMS server with a queue for the messages and then configure the JMS server using the Cloudera Manager Admin Console, as detailed below.

In addition to setting up and configuring the server side components, the policies (that include notifications about changes) are defined using the Cloudera Navigator console. See Using Policies to Automate Metadata Tagging for details about defining policies.

Configuring a JMS Server for Policy Messages

**Cloudera Manager Required Role:** Navigator Administrator (or Full Administrator)
Only the JMS URL, JMS User, and JMS Password settings are required to set up the JMS Server. **Enable Expression Input** only if you want to use Java expressions to define your policies. See [Metadata Policy Expressions](#) on page 48 for details about using Java expressions in your policy definitions.

These steps begin from the Cloudera Manager Admin Console home page.

1. **Select Clusters > Cloudera Management Service.**
2. **Click the Configuration tab.**
3. For choice of Filters, select **Category > Policies.** The configurable properties display:

![Configuration Properties](image)

4. **Set the following properties for each role group**

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JMS URL</td>
<td>tcp://localhost:61616</td>
<td>The URL of the JMS server. Notifications of changes to entities affected by policies you define are sent to this URL.</td>
</tr>
<tr>
<td>JMS User</td>
<td>admin</td>
<td>The JMS user to which notifications of changes to entities affected by policies are sent.</td>
</tr>
<tr>
<td>JMS Password</td>
<td>admin</td>
<td>The password of the JMS user to which notifications of changes to entities affected by policies are sent.</td>
</tr>
<tr>
<td>JMS Queue</td>
<td>Navigator</td>
<td>The JMS queue to which notifications of changes to entities affected by policies are sent.</td>
</tr>
<tr>
<td>Enable Expression Input</td>
<td>Disabled.</td>
<td>Click the box to enable if you want to use Java expressions to define policies. See <a href="#">Metadata Policy Expressions</a> on page 48 for details. When enabled, feature applies to the Navigator Metadata Server Default Group. If this configuration should apply to another role group (or groups), edit the value for the appropriate role group. See <a href="#">Modifying Configuration Properties Using Cloudera Manager</a>.</td>
</tr>
</tbody>
</table>

5. **Click Save Changes.**
6. **Restart the Navigator Metadata Server role instance.**
Cloudera Navigator and the Cloud

Market-leading organizations today are deploying clusters to the cloud to increase efficiency and reduce costs. Using public cloud infrastructure lets companies create optimal solutions that combine both on-premises and cloud clusters in persistent or transient deployment model. With Cloudera Altus, transient clusters are provisioned quickly, expand and shrink in response to varying workloads, and can be terminated just as easily when no longer needed. See Cloudera Enterprise in the Cloud for an overview.

In addition to full-scale cluster deployments to the cloud, on-premises Cloudera clusters can leverage cloud resources, such as Amazon Simple Storage Service (S3). For example, an Amazon S3 storage bucket can be used as the source or target for BDR (backup and disaster recovery), enabling organizations to replicate HDFS files and Hive data to and from the S3 bucket. As another example, an S3 bucket is often used as a persistent data store, with data copied to HDFS on a transient cluster for interactive or iterative workloads. Other supported cloud-based functionality includes using Amazon Simple Queue Service (SQS) to support notification from changes to Amazon S3 bucket objects.

Cloudera Navigator can collect metadata from clusters running on Cloudera Altus and from objects stored on Amazon S3. That means that the entities resulting from Hive jobs, for example, running on clusters instantiated by Cloudera Altus users can be seen in lineage diagrams in a centralized on-premises Cloudera Navigator instance running on a persistent or long-running Cloudera Manager cluster.

Note: The Cloudera Navigator instance runs in an on-premises or persistent cluster.

This chapter provides the details for setting up Cloudera Navigator to extract from cloud-based resources, including Cloudera Altus clusters and Amazon S3 buckets. After configuring support, data stewards, data engineers, and others use the Cloudera Navigator console to explore metadata, relationships, and trace data lineage to its source as with any other type of entity.

Using Cloudera Navigator with Altus Clusters

Cloudera Navigator can extract metadata and lineage from Cloudera Altus clusters. That means that data engineers running jobs on transient clusters and processing jobs using Hive on Altus clusters can view metadata and lineage in
Cloudera Navigator. Because Cloudera Navigator identifies cluster, cluster group, and other properties related to Altus deployed clusters, data engineers can use the lineage diagrams available in the Cloudera Navigator console to trace data back to its source, even when data processing spans different clusters.

Support for Cloudera Altus metadata and lineage extraction from Cloudera Navigator is not enabled by default. Configuration tasks must be completed in the Cloudera Altus console and in the Cloudera Manager Admin Console to enable this capability. Completing the configuration tasks requires your organization to have a Cloudera Altus account, and assume that you are successfully running jobs on clusters according to the processes detailed in the Cloudera Altus documentation.

Note: Cloudera Altus requires an account on the cloud service provider. For more information about Cloudera Altus and for a demonstration, visit Introducing Cloudera Altus. To request a Cloudera Altus account, go to the Altus registration page.

How it Works: Background to the Setup Tasks

Every cluster deployed using Cloudera Altus is deployed as a Cloudera Manager cluster, that is, it has its own Cloudera Manager Server. This Cloudera Manager Server instance is read-only, but an Altus user—for example, a data engineer developing a new Hive or MapReduce2 script—can use Cloudera Manager to view execution and other details. This Cloudera Manager Server also includes the Telemetry Publisher role instance (daemon) that publishes data from the transient cluster to other systems. The Telemetry Publisher stores metadata and lineage details from the running transient cluster to a cloud-based storage location (for example, an Amazon S3 bucket) specified in the Altus Environment used to instantiate the cluster.

It is the Telemetry Publisher role instance running in the Cloudera Manager Server of a Cloudera Altus cluster and the cloud-based storage mechanism that enables Cloudera Navigator to obtain metadata and lineage information from single-user transient clusters. Before the cluster shuts down, an internal process checks to ensure that all metadata and lineage from that transient cluster has been successfully published to the specified cloud-based storage resource.

Meanwhile, the on-premises Cloudera Navigator instance has been collecting the metadata and lineage information by reading from the same cloud-based storage location. Because cloud storage is a persistent store that runs independently and is not part of the transient cluster, it remains a viable source of data for Cloudera Navigator.

These system interactions are all but transparent to both Cloudera Altus users and Cloudera Navigator data stewards and other users. However, enabling the interaction involves several preliminary tasks across both Cloudera Altus and Cloudera Navigator. The specifics vary, depending on the cloud provider.

Configuring Extraction for Altus Clusters on AWS

Follow the steps below to configure Cloudera Navigator to extract metadata and lineage from single-user transient clusters deployed to Amazon Web Services using Cloudera Altus. The Cloudera Navigator extraction process for clusters launched by Cloudera Altus works as follows:

- Any HDFS paths in a job, query, or data entity are extracted as proxy entities for the path, similar to how Hive entities are extracted. That means that HDFS is not bulk extracted from an Altus cluster.
- Hive Metastore (HMS) entities are also not bulk extracted. Cloudera Navigator extracts Hive entities used in queries that generate lineage, such as databases, tables, and so on.

Requirements

Cloudera Navigator collects metadata and lineage entities from transient clusters deployed to AWS by Cloudera Altus users. The metadata and lineage data is not collected directly from the transient clusters but rather from an Amazon S3 bucket that serves as the storage mechanism for the Telemetry Publisher running in the cluster (see How it Works: Background to the Setup Tasks on page 127 for details).

As detailed in the steps below, successful integration of Cloudera Navigator and Cloudera Altus clusters depends on correctly:

- Identifying an Amazon S3 bucket for storing metadata
Configuring correct access permissions to the S3 bucket from both Cloudera Altus and Cloudera Navigator. Transient clusters instantiated by Altus users must have read and write permissions to the Amazon S3 bucket used by Telemetry Publisher. The on-premises centralized Cloudera Navigator instance must have read permissions on the same Amazon S3 bucket.

— Altus user account (with cross-account privileges to AWS account). Privileges to launch EC2 clusters and use other AWS resources, including Amazon S3 buckets identified in the Altus environment (data input, data output, logs, and the bucket for Telemetry Publisher).

— Read and write privileges to the Amazon S3 bucket configured in the Altus environment assigned to the Altus user.

— AWS access key ID and AWS secret key for the AWS account associated with the Amazon S3 bucket.

The steps below assume that you have:

- An Amazon Web Services account.
- A Cloudera Altus account.
- A Cloudera Altus user account that can run jobs on transient clusters deployed to AWS.
- Access to the on-premises or persistent Cloudera Manager cluster running Cloudera Navigator. The Cloudera Manager user role of Full Administrator and the ability to log in to the Cloudera Manager Admin Console is required.
- AWS Credentials for the AWS account hosting the Amazon S3 bucket that serves as the storage mechanism for metadata and lineage data from clusters on AWS launched by Cloudera Altus.

Obtaining AWS Credentials for the Amazon S3 Bucket

AWS Credentials are available to be downloaded whenever you create an IAM user account through the AWS Management Console. If you are configuring an existing Amazon S3 bucket and you do not have the AWS Credentials for it, you can generate new AWS Credentials from the AWS account using either the AWS Management Console or the AWS CLI.

**Important:** The AWS credentials must have read and write access to the Amazon S3 bucket.

Generating new AWS Credentials deactivates any previously issued credentials and makes the newly generated credentials Active for the AWS account. Keep that in mind if you obtain new AWS Credentials to use for the Cloudera Navigator-Cloudera Altus integration.
Note: If you have the AWS Credentials obtained when the account was created, do not regenerate a new set of AWS Credentials unless you want to change the credentials.

These steps assume you have an AWS account and that an Amazon S3 bucket exists on that account that you want to use as the storage location for metadata and lineage.

- Log in to the AWS Management Console using the account associated with the Amazon S3 bucket.
- Navigate to the Security credentials section of the Users page in IAM for this account. For example:

  ![AWS Security Credentials](image)

  - Click the Create access key button to generate new AWS Credentials. Extract the credentials (the Access Key Id and Secret Key) from the user interface or download the credentials.csv for later use.

  New credentials can be created by using the AWS CLI rather than the AWS Management Console. See Amazon documentation for details.

Cloudera Altus Configuration

Cloudera Altus instantiates single-user transient clusters focused on data engineering workloads that use compute services such as Hive or MapReduce2. The typical deployment scenario involves running scripts that invoke the Cloudera Altus CLI to instantiate the cluster, in this case, using Amazon Web Services according to the details specified in the Altus Environment. An Altus environment specifies all resources needed by the cluster, including the AWS account that will be used to instantiate the cluster. The Cloudera Altus user account is configured to provide cross-account access to the AWS account that has permissions to launch AWS Elastic Compute Cloud (EC2) instances and use other AWS resources, including Amazon S3 buckets.

Although the Altus Environment can be created using Quickstart, Cloudera recommends using the Environment Wizard or the Altus CLI instead. The environment wizard provides better control over configuring resources, including letting you specify the Amazon S3 bucket that clusters will use to store metadata and lineage information for collection by Cloudera Navigator. Specifically, the Instance Profile Role page of the Configuration Wizard lets you enable integration with Cloudera Navigator and specify the Amazon S3 bucket that will hold collected metadata and lineage information.

On the Instance Profile Role page of the Configuration Wizard, complete the following steps:

- Click the Enable checkbox for Cloudera Navigator Integration.
In the **Cloudera Navigator S3 Bucket** field, enter the path to the Amazon S3 bucket, including the final `/`, which identifies the target as an S3 bucket. For example:

```plaintext
s3a://cluster-lab.example.com/cust-input/
```

To provide the correct access to the S3 bucket, you must also create the appropriate policy in the AWS Management Console and apply the policy to the Amazon S3 bucket.

For more information about using Cloudera Altus, see the Cloudera Altus documentation.

**Cloudera Navigator Configuration**

The Cloudera Navigator runs in the context of Cloudera Manager Server. Its two role instances, the Navigator Audit Server and Navigator Metadata Server, run on the Cloudera Management Service. The Navigator Metadata Server role instance is the component that extracts metadata and lineage from the Amazon S3 bucket using the AWS Credentials configured for connectivity in the steps below:

- Follow the steps in Adding AWS Credentials and Configuring Connectivity on page 130 to add new or regenerated AWS Credentials to the Cloudera Manager Server and then configure connectivity.
- Follow the steps in Configuring Connectivity for AWS Credentials on page 132 to configure connectivity for AWS Credentials that are already available to be used for the Amazon S3 bucket but have not yet been configured for connectivity.

**Important:** Cloudera Navigator extracts metadata and lineage for clusters deployed using Altus from one Amazon S3 bucket only. In addition, for any given Amazon S3 bucket collecting metadata and lineage from Altus clusters, configure only one Cloudera Navigator instance to extract from that Amazon S3 bucket. Using multiple Cloudera Navigator instances to extract from the same Amazon S3 bucket is not supported and has unpredictable results.

**Adding AWS Credentials and Configuring Connectivity**

**Cloudera Manager Required Role:** **Full Administrator**

The AWS Credentials must be added to the Cloudera Manager Server for use by Cloudera Navigator. These credentials must be from the AWS account hosting the Amazon S3 bucket that is configured in the Altus environment.

**Note:** The AWS account associated with these credentials must have cross-account access permissions from the Altus user account that will launch clusters on AWS and run jobs. These credentials must also have read and write permissions on the S3 bucket because the clusters launched must be able to write metadata and lineage information to the Amazon S3 bucket as jobs run.

1. Log in to the Cloudera Manager Admin Console.
2. Select **Administration** > **AWS Credentials**.
3. Click the **Add Access Key Credentials** button on the AWS Credentials page.
   - Enter a meaningful name for the AWS Credential, such as the type of jobs the associated clusters will run (for example, **etl-processing**). This name is for your own information and is not checked against any Cloudera Altus or AWS attributes.
   - Enter the AWS Access Key ID and the AWS Secret Key.
4. Click **Add** to save the credentials. The S3Guard option page displays, reflecting the credential name (for example, **Edit S3Guard: etl-processing**). Disregard this option.

5. Click **Save**. The Connect to Amazon Web Services page displays, showing the options available for this specific AWS credential.

6. Click **Enable Metadata and Lineage Extraction from Cloudera Altus**. The Metadata and Lineage Extraction Configuration setting page displays a field for specifying the Amazon S3 bucket name.

7. Enter the name of the Amazon S3 bucket. For example:

8. Click **OK**. The AWS Credentials page re-displays, and the newly added AWS Credential is listed with any other AWS Credentials held by the Cloudera Manager Server.
9. Restart the Cloudera Management Service.

When the service restarts, the AWS credentials will be used by Cloudera Navigator to authenticate to the AWS account and extract metadata and lineage stored on the specified S3 bucket.

Configuring Connectivity for AWS Credentials

If the AWS Credentials are available for the Amazon S3 bucket, you can configure them as follows:

1. Log in to the Cloudera Manager Admin Console.
2. Select Administration > AWS Credentials.
3. Find the available AWS Credentials that provide access to the Amazon S3 bucket used to collect metadata and lineage from transient clusters.

AWS Credentials

AWS Credentials allow CDH services and Cloudera tools to securely query data, browse data, backup and restore data/metadata, search metadata and view data lineage of data in Amazon S3. More Details

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Connectivity</th>
<th>Creation</th>
<th>Last Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>data-eng-s3-nav-bucket</td>
<td>Access Key Credentials</td>
<td>September 15, 2017 8:14 PM</td>
<td>September 15, 2017 8:14 PM</td>
<td></td>
</tr>
<tr>
<td>cust-data</td>
<td>Access Key Credentials</td>
<td>September 15, 2017 8:15 PM</td>
<td>September 15, 2017 8:15 PM</td>
<td></td>
</tr>
<tr>
<td>offside-hdfs-backup</td>
<td>Access Key Credentials</td>
<td>Cloudera Navigator</td>
<td>September 15, 2017 8:15 PM</td>
<td>September 15, 2017 8:15 PM</td>
</tr>
</tbody>
</table>

4. Click the Actions drop-down menu and select Edit Connectivity. The Connect to Amazon Web Services page displays the three sections of possible configurations.

5. In the Cloudera Navigator section, click the Enable Metadata and Lineage Extraction from Cloudera Altus link. The Metadata and Lineage Extraction Configuration page displays.

6. Enter the name of the Amazon S3 bucket in the S3 Bucket Name field.

7. Click OK.

8. Restart the Cloudera Management Service.

This completes the setup process. After the restart, metadata and lineage for transient clusters deployed using Cloudera Altus should be available in the Cloudera Navigator console.

Note: See Troubleshooting to identity possible issues if metadata and lineage do not display in the Cloudera Navigator console after completing the configuration and restarting the system.

Technical metadata specific to clusters deployed using Altus include the following property names and types:

- Cluster (Source Type)
- Cluster-name (Cluster Group)
- Transient (Deployment Type)
- Cluster Template, Cluster Instance (Classname)

For example:
Using Cloudera Navigator with Amazon S3

Amazon Simple Storage Service (S3) is a storage solution offered by Amazon Web Services (AWS) that provides highly available storage in the cloud. Clusters deployed not only in the AWS cloud but also on-premises are using Amazon S3 as persistent storage. Common use cases include BDR (backup and disaster recovery) and persistent storage for transient clusters deployed to the cloud, such as storage for ETL workload input and output.

As with data stored on HDFS and processed using compute engines like Hive and Impala, Cloudera Navigator can obtain metadata and lineage from Amazon S3 storage. There are some limitations and constraints as discussed below, and some setup is required to enable this capability (see Configuring Extraction for Amazon S3 on page 137).

Note: Cloudera Navigator does not audit Amazon S3 storage buckets. Only Cloudera Navigator metadata and lineage are supported.

This section provides conceptual information about Amazon S3 storage and shows you how to configure Cloudera Navigator to extract metadata and lineage from an Amazon S3 bucket.

Continue reading:

- Amazon S3 Storage Characteristics on page 133
- Cloudera Navigator and Amazon S3 on page 134
  - Object Lifecycle Rules Constraints on page 134
  - Amazon SQS and Amazon SNS Constraints on page 134
  - Object Storage and Implicit Folder Limitation on page 135
- Overview of Amazon S3 Extraction Processes on page 136
  - API Usage and Setting Limits on page 136

Amazon S3 Storage Characteristics

Amazon S3 is an object store rather than a file store or block store. It does not have the hierarchy found in typical filesystems. Amazon S3 uses the construct of a bucket as a container for objects. An object can be any kind of file—text file, image, photo, graphic, video, an ETL bundle to be ingested into a cluster, and so on.

Files can be added to Amazon S3 through the AWS Management Console, by using the AWS CLI, or by using scripts that invoke the CLI.

Amazon S3 storage is highly available because Amazon replicates data across multiple servers within its data centers and uses an eventual consistency model—not all accesses of an object on Amazon S3 may be reflected concurrently or instantaneously. However, eventually, all updates to data across servers are synchronized. The eventual consistency
model can result in a short delay between the time objects are uploaded to Amazon S3 and the time their metadata is available in Cloudera Navigator. This is expected behavior and simply how eventual consistency works.

Note: The eventual consistency model of Amazon S3 can be augmented by using S3Guard, which leverages Amazon DynamoDB to provide support for transactions. See Configuring and Managing S3Guard in the Cloudera Administration guide for details.

For more information about Amazon S3, see Amazon S3 documentation.

Cloudera Navigator and Amazon S3

Despite the implementation details of Amazon S3 storage, Cloudera Navigator collects metadata for Amazon S3 entities in much the same way as for HDFS entities, with some exceptions shown in the table below.

Note: In addition to metadata, Cloudera Navigator extracts lineage from Hive, Impala, and MapReduce (except for MapReduce Glob paths) on Amazon S3.

The following table lists some differences between object types and supported features offered by Amazon S3 and how those are supported by Cloudera Navigator:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Amazon S3</th>
<th>Cloudera Navigator</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-defined metadata</td>
<td>✓</td>
<td>◯</td>
</tr>
<tr>
<td>consists of custom key-value pairs (in which each key is prefixed with x-amz-meta-) that can be used to tag objects on Amazon S3.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System-defined metadata</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>includes properties such as Date, Content-Length, Last-Modified. Some system-defined properties comprise the Technical Metadata for the object in Cloudera Navigator.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tags for buckets and objects</td>
<td>✓</td>
<td>◯</td>
</tr>
<tr>
<td>Versioning is not supported. Cloudera Navigator extracts metadata and lineage from the latest version only.</td>
<td>✓</td>
<td>◯</td>
</tr>
<tr>
<td>Unnamed directories</td>
<td>✓</td>
<td>◯</td>
</tr>
<tr>
<td>Object lifecycle rules. See Object Lifecycle Rules Constraints on page 134 for more information.</td>
<td>✓</td>
<td>◯</td>
</tr>
<tr>
<td>Amazon Simple Queue Service (SQS). See Amazon SQS and Amazon SNS Constraints for usage limitations and requirements.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Amazon Simple Notification Service (SNS). See Amazon SQS and Amazon SNS Constraints for usage limitations and requirements.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Object Lifecycle Rules Constraints

Cloudera Navigator does not support lifecycle rules that remove objects from Amazon S3. For example, an object lifecycle rule that removes objects older than $n$ days deletes the object from Amazon S3 but the event is not tracked by Cloudera Navigator. This limitation applies to removing objects only. Using lifecycle rules requires using bulk-only extraction. See Custom Configurations on page 139 for details about configuring the necessary AWS Policy and applying it to the Amazon S3 bucket for use by Cloudera Navigator.

Amazon SQS and Amazon SNS Constraints

Amazon Simple Queue Service (SQS) is a distributed, highly scalable hosted queue for storing messages. Amazon Simple Notification Service (SNS) is publish-subscribe notification service that coordinates message delivery. Both services can be configured for use with Amazon S3 storage buckets. For example, Amazon S3 storage buckets can send notification
messages to one or more queues or to email addresses whenever specified events occur, such as creating, renaming, updating, or deleting data on the Amazon S3 bucket.

During the default configuration process, Cloudera Navigator transparently sets up an Amazon SQS queue and configures Amazon S3 event notification for each bucket. The queue is used to hold event messages that are subsequently collected by the Cloudera Navigator S3 extractor process, for incremental extracts. Use the default configuration process only for Amazon S3 buckets that do not have existing queues or notifications configured.

For Amazon S3 buckets that are already configured for queues, use the custom configuration process—sometimes referred to as "Bring Your Own Queue" (BYOQ)—to manually configure queues for Cloudera Navigator. For Amazon S3 buckets that are already configured for notifications, use the BYOQ custom configuration in conjunction with Amazon SNS in a fan-out configuration. In a fan-out scenario, an Amazon SNS message is sent to a topic and then replicated and pushed to multiple Amazon SQS queues, HTTP endpoints, or email addresses. See Common Amazon SNS Scenarios for more information about fan-out configuration, and see Custom Configurations on page 139 for details about configuring Cloudera Navigator when the Amazon S3 bucket is already set up for either Amazon SQS or Amazon SNS.

Object Storage and Implicit Folder Limitation

Amazon S3 storage does not use a directory structure or other hierarchy as found in a traditional file system. Each object has an object key name that identifies the object by its S3Uri location—the path to the object. This path includes the object, prefix if any, and bucket name. Including the S3 protocol specifier, the pattern is as follows:

`s3://bucketname/prefix/objectkey`

There can be more than one prefix in an object key name. Prefixes are separated by the forward slash character (/). Although Amazon S3 provides a folder metaphor for organizing objects in an S3 bucket, the folder does not provide actual containment or structure: it is the object key name and its S3Uri location that identifies the object.

Cloudera Navigator mimics file system behavior by mapping elements of the object key name to implicit folders. For example, for an Amazon S3 file with the object key name `2017/08_AUG/data-raw.csv`, Cloudera Navigator creates an entity with the path `2017/08_AUG/data-raw.csv` and also creates two directories: `2017` and `2017/08_AUG`.

---

<table>
<thead>
<tr>
<th>Cloudera Navigator</th>
<th>Amazon S3</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Technical Metadata" /></td>
<td><img src="image" alt="Amazon S3" /></td>
</tr>
</tbody>
</table>

Cloudera Navigator console Lineage tab for the file with object key `2017/08_AUG/data-raw.csv` shows it in the context of implicit folders:
Cloudera Navigator has some limitations specifically for deleted objects and implicit folders as follows:

- Cloudera Navigator does not mark an implicit folder as deleted even after all its child objects have been deleted.
- Cloudera Navigator does not mark as deleted any objects and folders deleted using Amazon S3 tools, such as the AWS CLI (`aws s3 commands`) or the AWS Management Console.

**Note:** To filter out implicit folders from the S3 entities displayed, enter `implicit:false` in the Search field. Conversely, to find implicit entities enter `implicit:true` in the Search field.

For more details about the properties shown by Cloudera Navigator, see [S3 Properties](page 167) in the Reference section of this guide.

Despite the differences between an object store and a hierarchical store, data engineers can work with Amazon S3 using the Cloudera Navigator in much the same way as for HDFS and other entities.

**Overview of Amazon S3 Extraction Processes**

By default, Cloudera Navigator uses combined bulk and incremental extraction processes. An initial bulk process extracts all metadata from an Amazon S3 bucket during the configuration process. Subsequent extracts are incremental. Changes are collected from an Amazon SQS queue created by Cloudera Navigator during the default configuration process.

This bulk-plus-incremental extraction combination provides the optimal performance for production systems and is also the most cost-effective in terms of Amazon API usage:

- For the bulk extract, Cloudera Navigator invokes the Amazon S3 API.
- For the incremental extract, Cloudera Navigator invokes the Amazon SQS API.

Amazon meters usage and charges differently for each of these APIs.

**API Usage and Setting Limits**

Amazon bills on a monthly basis and resets the billing cycle each month. To help manage the monthly cost of using these APIs, Cloudera Navigator provides a [safety valve property](#) that can be set to limit its use of the AWS APIs. If you decide to configure this property to set a limit on API usage, keep the following in mind:

- If the limit is set is reached in any given 30-day interval, Cloudera Navigator suspends extraction from the configured Amazon S3 buckets until the next 30-day interval begins.
- When the new 30-day interval begins, Cloudera Navigator extracts any data that was not extracted while extraction was suspended.

To set a limit on the AWS API usage:

- Use Cloudera Manager Admin Console to access the [Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties](#).
- Set the value of `any_int` to your chosen limit.

````
nav.aws.api.limit=any_int
```
See Setting Properties with Advanced Configuration Snippets on page 143 for details about using Cloudera Manager Admin Console for this task.

Note: Cloudera Navigator does not notify you if extraction is ever suspended due to the limit you have set in the safety valve. Cloudera recommends setting a billing alarm in the AWS Management Console to get notified by Amazon when your monthly use of the APIs exceeds the limit you have set in the safety valve.

Configuring Extraction for Amazon S3

Depending on the specifics of the Amazon S3 bucket targeted for extraction by Cloudera Navigator, the configuration process follows one of two alternative paths:

- **Default configuration**—The default configuration is available for Amazon S3 buckets that have no existing Amazon SQS or Amazon SNS services configured. During configuration, Cloudera Navigator accesses the configured AWS account, performs an initial bulk extract from the Amazon S3 bucket, sets up Amazon SQS queues in each region with buckets, and sets up event notifications for each bucket for subsequent incremental extracts—all transparently to the Cloudera Manager administrator handling the configuration process.

- **Custom configuration**—Custom configuration is required for any Amazon S3 bucket that is currently using Amazon SQS (has queues set up for other applications, for example) or is setup for notifications using Amazon SNS. In these cases you must manually configure a new queue—**bring your own queue**—and in some cases, additionally configure Amazon SNS for **fanout**.

Continue reading:

- AWS Credentials Requirements on page 137
- Default Configuration on page 138
- Custom Configurations on page 139
  - Configuring Your Own Queues on page 139
    - Configure the Queue for Cloudera Navigator on page 140
    - Configure Event Notification for the Queues on page 141
    - Configuring Amazon SNS Fan-out on page 141
  - Defining and Attaching Policies on page 142
    - Event Notification Policy for Custom Queues on page 142
    - Extraction Policies for Custom Queues on page 142
    - Extraction Policies JSON Reference on page 142
  - Setting Properties with Advanced Configuration Snippets on page 143
  - Cloudera Navigator Properties for Amazon S3 on page 144

AWS Credentials Requirements

Cloudera Manager can have multiple AWS credentials configured for various purposes at any given time. These are listed by name on the AWS Credentials page which is accessible from the Cloudera Manager Admin Console, under the Administration menu. However, there are specific constraints on AWS credentials for Cloudera Navigator as follows:

- Navigator supports a single key for authentication; only one AWS credential can be used for a given Navigator instance. Navigator can extract metadata from any number of S3 buckets, assuming the buckets can be accessed with the configured credential.
- An AWS credential configured for connectivity from one Cloudera Navigator instance cannot be used by another Cloudera Navigator instance. Configuring the same AWS credentials for use with different Cloudera Navigator instances can result in unpredictable behavior.
- Cloudera Navigator requires an AWS credential associated with an IAM user identity rather than an IAM role.
- Any changes to the AWS credentials (for example, if you rotate credentials on a regular basis) must be for the same AWS account (IAM user). Changing the AWS credentials to those of a different IAM user results in errors.
from the Amazon Simple Queue Service (used transparently by Cloudera Navigator). If a new key is provided to Cloudera Navigator, the key must belong to the same AWS account as the prior key.

- For the default configuration, the account for this AWS credential must have administrator privileges for:
  - Amazon S3
  - Amazon Simple Queue Service (SQS)

- For the custom configurations, the account needs privileges for Amazon S3, Amazon SQS, and for Amazon Simple Notification Service (SNS).

Default Configuration

The steps below assume that you have the required AWS credentials for the IAM user with the Amazon S3 bucket. Amazon Web Services (AWS) account (an IAM user account) and that you can use the AWS Management Console. The AWS credentials for the IAM user are configured for Cloudera Navigator using the Cloudera Manager Admin Console during the configuration process below.

**Important:** If the Amazon S3 bucket is already configured for queuing or notification, do not follow the steps in this section. See Custom Configurations on page 139 instead.

Configuring Cloudera Navigator for Amazon S3

At the end of the configuration process detailed below, Cloudera Navigator authenticates to AWS using the credentials and performs an initial bulk extract of entities from the Amazon S3 bucket. It also sets up the necessary Amazon SQS queue (or queues, one for each region) to use for subsequent incremental extracts.

The steps below assume you have the required AWS credentials available.

1. Log in to the Cloudera Manager Admin Console.
2. Click **Administration > AWS Credentials**. The AWS Credentials page displays, listing any existing credentials that have been setup for the Cloudera Manager cluster.
3. Click the **Add Access Key Credentials** button. In the Add Access Key Credentials page:
   a. Enter a **Name** for the credentials. The name can contain alphanumeric characters and can include hyphens, underscores, and spaces but should be meaningful in the context of your production environment. Use the name of the Amazon S3 bucket or its functionality, for example, *cust-data-raw* or *post-proc-results*.
   b. Enter the **AWS Access Key ID**.
   c. Enter the **AWS Secret Key**.
4. Click **Add**. The **Edit S3Guard:aws-cred-name** page displays, giving you the option to enable S3Guard for the S3 bucket.
- Click the Enable S3Guard box only if the AWS credential has privileges on Amazon DynamoDB and if the preliminary S3Guard configuration is complete. See Cloudera Administration for details about Configuring and Managing S3Guard.

5. Click Save.

**Note:** Cloudera Manager stores the AWS credential securely in a non-world readable location. The access key ID and secret values are masked in the Cloudera Manager Admin Console, encrypted before being passed to other processes, and redacted in the logfiles.

The Connect to Amazon Web Services page displays the credential name and services available for its use:

6. Click the Enable S3 Metadata Extraction link in the Cloudera Navigator section of the page. A Confirm prompt displays, notifying you that Cloudera Navigator must be manually restarted after this change.

7. Click OK to enable the connection.

8. At the top of the Cloudera Manager Admin Console, click the Stale Configuration restart button when you are ready to restart Cloudera Navigator.

Metadata and lineage for Amazon S3 buckets will be available in the Cloudera Navigator console along with other sources, such as HDFS, Hive, and so on. It may take several minutes to complete the initial extraction depending on the number of objects stored on the Amazon S3 bucket.

**Custom Configurations**

Follow these steps for Amazon S3 buckets that are already configured with queues or event notifications. Custom configurations include coconfiguring your own queue (BYOQ) and BYOQ with Fan-out, as detailed below.

**Configuring Your Own Queues**

Sometimes referred to as Bring Your Own Queue (BYOQ), configuring your own queue is required if the Amazon S3 bucket being targeted for extraction by Cloudera Navigator already has existing queues or is configured for notifications. The process involves stopping Cloudera Navigator and then using the AWS Management Console for the following tasks:
Creating and configuring an Amazon Simple Queue Service (SQS) queue for Cloudera Navigator for each region in which the AWS (IAM user) account has Amazon S3 buckets.

- Configuring Amazon Simple Notification Service (SNS) on each bucket to send Create, Rename, Update, Delete (CRUD) events to the Cloudera Navigator queue.
- Configuring the bucket for Notification Fan-Out if needed to support existing notifications configured for other applications.
- Adding a Policy for the appropriate extraction process (Bulk + Incremental, Bulk Only) to the IAM user account.
- Adding the Policy for event notifications to the IAM user account.

Important: Always make sure any newly created Amazon S3 buckets are configured for event notifications before adding data so the queues are properly updated.

Configure the Queue for Cloudera Navigator
This manual configuration process requires stopping Cloudera Navigator. You must create a queue for each region that has S3 buckets.

1. Log in to Cloudera Manager Admin Console and stop Cloudera Navigator:
   - Select Clusters > Cloudera Management Service
   - Click the Instances tab.
   - Click the checkbox next to Navigator Audit Server and Navigator Metadata Server in the Role Type list to select these roles.
   - From the Actions for Selected (2) menu button, select Stop

2. Log in to the AWS Management Console with AWS account (IAM user) and open the Simple Queue Service setup page (select Services > Messaging > Simple Queue Service. Click Create New Queue or Get Started Now if region has no configured queues.)

3. For each region that has Amazon S3 buckets, create a queue as follows:
   a. Click the Create New Queue button. Enter a Queue Name, click the Standard Queue (not FIFO), and then click Configure Queue. Configure the queue using the following settings:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default Visibility Timeout</td>
<td>10 minutes</td>
</tr>
<tr>
<td>Message Retention Period</td>
<td>14 days</td>
</tr>
<tr>
<td>Delivery Delay</td>
<td>0 seconds</td>
</tr>
<tr>
<td>Receive Message Wait Time</td>
<td>0 seconds</td>
</tr>
</tbody>
</table>

   b. Select the queue you created, click the Permissions tab, click Add a Permission, and configure the following in the Add a Permission to... dialog box:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effect</td>
<td>Allow</td>
</tr>
<tr>
<td>Principal</td>
<td>Everybody</td>
</tr>
<tr>
<td>Actions</td>
<td>SendMessage</td>
</tr>
</tbody>
</table>

   c. Click the Add Conditions (optional) link open the condition fields and enter the following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qualifier</td>
<td>None</td>
</tr>
<tr>
<td>Condition</td>
<td>ArnLike</td>
</tr>
<tr>
<td>Key</td>
<td>aws:SourceArn</td>
</tr>
<tr>
<td>Value</td>
<td>arn:aws:s3:::<em>::</em></td>
</tr>
</tbody>
</table>
d. Click **Add Condition** to save the settings.

e. Click **Add Permission** to save all settings for the queue.

Repeat this process for each region that has Amazon S3 buckets.

### Configure Event Notification for the Queues

After creating queues for all regions with Amazon S3 buckets, you must configure event notification for each Amazon S3 bucket. Assuming you are still logged into the AWS Management Console:

1. Navigate to the Amazon S3 bucket for the region (Services > Storage > S3).
2. Select the bucket.
3. Click the **Properties** tab.
4. Click the **Events** settings box.
5. Click **Add notification**.
6. Configure event notification for the bucket as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>nav-send-metadata-on-change</th>
</tr>
</thead>
</table>
| Events | • ObjectCreated(All)  
• ObjectRemoved(All) |
| Send to | SQS Queue |
| SQS queue | Enter the name of your queue |

**Important:** Cloudera Navigator extracts metadata from one queue only for each region.

### Configuring Amazon SNS Fan-out

Configure SNS fanout if you have existing S3 event notification. For more information about SNS fanout, see Amazon documentation for **Common SNS Scenarios**.

1. Add the following to the **Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties**. See [Setting Properties with Advanced Configuration Snippets](#) on page 143 for details about using Cloudera Manager Admin Console if necessary.

```java
nav.s3.extractor.incremental.enable=true
nav.s3.extractor.incremental.auto_setup.enable=false
nav.s3.extractor.incremental.queues=queue_json
```

Specify the queue properties using the following JSON template (without any spaces). Escape commas (,) by preceding them with two backslashes (\), as shown in the template:

```json
{"region":"us-west-1","queueUrl":"https://sqs.aws_region.amazonaws.com/account_num/queue_name"}, {queue_2}
```
2. Restart Cloudera Navigator.

**Defining and Attaching Policies**

**Event Notification Policy for Custom Queues**

To enable event notification for a custom queue, create the following policy by copying the policy text and pasting it in the policy editor, and then attaching it to the Cloudera Navigator user in AWS.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Stmt1481678612000",
         "Effect": "Allow",
         "Action": [
            "sqs:DeleteMessage",
            "sqs:DeleteMessageBatch",
            "sqs:GetQueueAttributes",
            "sqs:ReceiveMessage"
         ],
         "Resource": "*"
      },
      {
         "Sid": "Stmt1481678744000",
         "Effect": "Allow",
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListAllMyBuckets",
            "s3:ListBucket",
            "s3:GetObject",
            "s3:GetObjectAcl",
            "s3:GetBucketNotification",
            "s3:PutBucketNotification"
         ],
         "Resource": ["arn:aws:s3:::*"]
      }
   ]
}
```

**Extraction Policies for Custom Queues**

Custom configurations require a valid extraction policy be defined and attached to the AWS user account associated with the Amazon S3 bucket. The policy is a JSON document that specifies the type of extraction. As mentioned in *Overview of Amazon S3 Extraction Processes* on page 136, the two types of extraction are as follows:

- Bulk + Incremental—This is the recommended approach for both cost and performance reasons and is used by the Default Configuration process automatically.
- Bulk Only—This approach is recommended for proof-of-concept deployments. It is required for the BYOQ with Fan-out configuration. In addition to applying the policy as detailed below, this approach also requires setting the nav.s3.extractor.incremental.enable property to false. See *Setting Properties with Advanced Configuration Snippets* on page 143 and the Cloudera Navigator Properties for Amazon S3 on page 144 for details.

**To configure the policy:**

- Log in to the AWS Management Console using the IAM user account associated with the target Amazon S3 bucket.
- Copy the appropriate JSON text from the table below *Extraction Policies JSON Reference* on page 142 and paste into the AWS Management Console policy editor for the Navigator user account on (the IAM user) through the AWS Management Console.

**Extraction Policies JSON Reference**
<table>
<thead>
<tr>
<th><strong>Bulk + Incremental (Recommended)</strong></th>
<th><strong>Bulk Only</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Initial bulk process extracts all metadata.</td>
<td>• Must be used for Amazon S3 buckets configured for event notifications.</td>
</tr>
<tr>
<td>• Subsequent incremental process extracts changes only (CRUD).</td>
<td></td>
</tr>
<tr>
<td>• Cannot be used with Amazon S3 buckets that use event notification.</td>
<td></td>
</tr>
</tbody>
</table>

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1481678614000",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "sqs:CreateQueue",
        "sqs:DeleteMessage",
        "sqs:DeleteMessageBatch",
        "sqs:GetQueueAttributes",
        "sqs:GetQueueUrl",
        "sqs:ReceiveMessage",
        "sqs:SetQueueAttributes"
      ],
      "Resource": "*"
    },
    {
      "Sid": "Stmt1481678744000",
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:ListAllMyBuckets",
        "s3:ListBucket",
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:GetBucketNotification",
        "s3:PutBucketNotification"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}
```

**Setting Properties with Advanced Configuration Snippets**

Certain features require additional settings or changes to the Cloudera Navigator configuration. For example, configuring BYOQ queues to use bulk-only extraction requires not only creating and attaching the extraction policy but also adding the following snippet to the Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties setting:

```
nav.s3.extractor.incremental.enable=false
```

To change property values by adding an advanced configuration snippet:

- Log in to the Cloudera Manager Admin Console.
- Select Clusters > Cloudera Management Service.
- Click Configuration.
Click **Navigator Metadata Server** under the Scope filter, and click **Advanced** under the Category filter.

- Enter **Navigator Metadata Server Advanced Configuration Snippet (Safety Valve)** for cloudera-navigator.properties in the Search field to find the property.
- Enter the property and its setting as a key-value pair, for example:

```
property=your_setting
```

- Click **Save Changes**.
- Restart the Navigator Metadata Server instance.

**Cloudera Navigator Properties for Amazon S3**

The table below lists the Navigator Metadata Server properties (cloudera-navigator.properties) that control extraction and other features related to Amazon S3. These properties can be set using the Cloudera Manager Admin Console to set properties in the advanced configuration settings.

Changing any of the values in the table requires a restart of Cloudera Navigator.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nav.aws.api.limit</td>
<td>Default is 5,000,000. Maximum number of Amazon Web Services (AWS) API calls that Cloudera Navigator can make per month.</td>
</tr>
<tr>
<td>nav.sqs.max_receive_count</td>
<td>Default is 10. Number of retries for inconsistent SQS messages (inconsistent due to eventual consistency).</td>
</tr>
<tr>
<td>nav.s3.extractor.enable</td>
<td>Default is true when an AWS credential has been configured to extract metadata from Amazon S3.</td>
</tr>
<tr>
<td>nav.s3.extractor.incremental.auto_setup.enable</td>
<td>Default is true. Enables Cloudera Navigator to set up Amazon SQS queues to receive notifications from Amazon S3 events. Set to false to disable the automatic setup and custom configure your own queue (BYOQ with Fan-out).</td>
</tr>
<tr>
<td>nav.s3.extractor.incremental.batch_size</td>
<td>Default is 1000. Number of messages held in memory during the extraction process.</td>
</tr>
<tr>
<td>nav.s3.extractor.incremental.enable</td>
<td>Default is true. Enables incremental extraction. Setting to false disables incremental extraction and effectively enables bulk-only extraction.</td>
</tr>
<tr>
<td>nav.s3.extractor.incremental.event.overwrite</td>
<td>Default is false. Prevents any existing event notifications from being overwritten by Cloudera Navigator auto-generated queues created during default configuration process.</td>
</tr>
<tr>
<td>nav.s3.extractor.incremental.queues</td>
<td>No default queue. Used by the custom configuration only. Specify a list of queues for the custom configuration using the JSON template. The list of queues should include the existing queues already in use and the newly configured queue that Cloudera Navigator will use for incremental extracts.</td>
</tr>
<tr>
<td>nav.s3.extractor.max_threads</td>
<td>Default is 3. The number of extractors (worker processes) to run in parallel.</td>
</tr>
<tr>
<td>nav.s3.home_region</td>
<td>Default is us-west-1. AWS region closest to the cluster and the Cloudera Navigator instance. Select the same AWS region (or the nearest one geographically) to minimize latency for API requests.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>nav.s3.implicit.batch_size</td>
<td>Default is 1000. Number of Solr documents held in memory when updating the state of implicit directories.</td>
</tr>
</tbody>
</table>
Cloudera Navigator APIs

Cloudera Navigator exposes REST APIs on the host running the Navigator Metadata Server role instance. In-house, third-party, and other developers can use the APIs to provide functionality not currently available through the Cloudera Navigator console or to customize behavior.

The APIs are used by the Cloudera Navigator console at runtime as users interact with the system. The Cloudera Navigator console has a debug mode that captures API calls and responses to a file that can then be downloaded and analyzed, not only for development projects but also for troubleshooting. Several examples of direct use of the APIs are included in this section.

Navigator APIs Overview

Documentation and a brief tutorial for the Navigator APIs are available on the same node of the cluster that runs the Navigator Metadata Server role. These are also accessible from the Cloudera Navigator console. Example pathnames for direct access (outside the Cloudera Navigator console) are shown in the table.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default location</th>
</tr>
</thead>
<tbody>
<tr>
<td>APIs</td>
<td><a href="http://fqdn-n.example.com:port/api/vn/operation">http://fqdn-n.example.com:port/api/vn/operation</a></td>
</tr>
<tr>
<td>API Documentation</td>
<td><a href="http://fqdn-n.example.com:port/api-console/index.html">http://fqdn-n.example.com:port/api-console/index.html</a></td>
</tr>
<tr>
<td>API Usage Tutorial</td>
<td><a href="http://fqdn-n.example.com:port/api-console/tutorial.html">http://fqdn-n.example.com:port/api-console/tutorial.html</a></td>
</tr>
</tbody>
</table>

Operations on the API are invoked using the following general pattern:

```
http://fqdn-n.example.com:port/api/vn/operation
```

The \( n \) in \( vn \) for the APIs represents the version number, for example, v11 for the Cloudera Navigator version 2.11.x. The API version number is located at the bottom of the Cloudera Navigator API documentation page.

The API uses HTTP Basic Authentication and accepts the same users and credentials that have access to the Cloudera Navigator console.

The resources listed in the table are aimed at technical and general audiences for Cloudera Navigator Data Management. The GitHub repository is aimed at developers.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default location</th>
</tr>
</thead>
<tbody>
<tr>
<td>GitHub Cloudera/navigator-sdk</td>
<td>Cloudera Navigator SDK is a client library that provides functionality to help users extract metadata from Navigator and to enrich the metadata in Navigator with custom metadata models, entities, and relationships.</td>
</tr>
<tr>
<td>Cloudera Community Data Management forum</td>
<td>Cloudera Community &gt; Data Management &gt; Data Discovery, Optimization forum for Cloudera Navigator users and developers.</td>
</tr>
<tr>
<td>Cloudera Engineering Blog</td>
<td>In-depth technical content provided by Cloudera engineers about components and specific subject areas.</td>
</tr>
</tbody>
</table>

Accessing API Documentation

The API documentation is available from the Cloudera Navigator console from a menu selection that navigates to the same URL listed in the table above.

To access the API documentation and tutorial:
1. Click Help.
2. Select **API Documentation**. The interactive API documentation page displays in a new window.
3. Click the Tutorial link near the top of the page to view the Cloudera Navigator API Usage Tutorial to learn about the APIs.

**Locating the Version Information for the Navigator API**

1. Click Help.
2. Select **API Documentation** to display the API documentation.
3. Scroll to the bottom of the page. The version number is the last item on this page:

   [BASE URL: http://node-5.example.com:7187/api/api-docs, API VERSION: v10]

The API is structured into resource categories. Click a category to display the resource endpoints.

**Mapping API Versions to Product Versions**

The following table identifies the compatibility between API versions and Cloudera Manager and Cloudera Navigator product versions.

<table>
<thead>
<tr>
<th>Cloudera Manager</th>
<th>Cloudera Navigator</th>
<th>Navigator API</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.14 – 5.16 and later</td>
<td>2.13 – 2.15 and later</td>
<td>v13</td>
</tr>
<tr>
<td>5.13</td>
<td>2.12</td>
<td>v12</td>
</tr>
<tr>
<td>5.12</td>
<td>2.11</td>
<td>v11</td>
</tr>
<tr>
<td>5.10 – 5.11</td>
<td>2.9 – 2.10</td>
<td>v10</td>
</tr>
<tr>
<td>5.7 – 5.9</td>
<td>2.6 – 2.8</td>
<td>v9</td>
</tr>
<tr>
<td>5.6</td>
<td>2.5</td>
<td>v8</td>
</tr>
</tbody>
</table>

**Using Debug Mode**

Cloudera Navigator console debug mode captures API responses and requests during interaction with the server. Use debug mode to facilitate development efforts or when asked by Cloudera Support to troubleshoot issues.

**Tip:** Exploring debug files can be a good way to learn about the APIs.

The general process for obtaining debug files to send to Cloudera Support is as follows:

1. Navigate to the Cloudera Navigator console menu item or page for which you want to obtain debug information.
2. Enable debug mode.
3. Reload the page and perform the specific action. For example, if you need to send a debug file to Cloudera Support for an error message that displays when you click on an entity in one of the charts, perform that action to raise the error message.
4. Download the debug file or files. Each time the counter changes, the content of the debug file content changes, so you should download each time the counter updates.
5. Disable debug mode.

To follow these steps you must be **logged in to the Cloudera Navigator console**.

**Enable Debug Mode**

The Enable Debug Mode menu selection is effectively a toggle. In other words, when you Enable Debug Mode, the menu selection changes to Disable Debug Mode. To enable debug mode:

1. Click the **user_name** from the menu to open the drop-down menu.
2. Select **Enable Debug Mode**. The Cloudera Navigator consoles displays a notification message in the bottom right of the page:
Notice the links to Disable and Download debug file, which you can use at any time to do either. Do not disable debug mode until you have downloaded the debug file you need.

3. Reload the browser page.

The counter \( n \) displays the number of API calls captured in the debug file for the current session interaction. Here is an example from a downloaded debug file showing one of the responses after reloading the page:

```
{
    "type": "GET",
    "url": "/api/v1/dashboard/clusters",
    "status": 200,
    "responseText": "{"\"clusterInfo\": [{\"\"clusterId\": \"Cluster 1\"},
                      {\"\"clusterDisplayText\": \"Cluster 1\"}]},
    "page": "http://fqdn-1.example.com:7187/dashboard.html",
    "timestamp": 1497719524407
}
```

**Download Debug File or Files**

As you perform actions with the Cloudera Navigator console, the counter refreshes and keeps tracks of current calls only (stateless nature of REST). In other words, download the debug file for any given action or selection that you want to examine.

**To download the debug file:**

- Click the *Download debug file* link in the debug-enabled notification message.
- Save the file to your local workstation.

Debug files use a naming convention that includes the prefix "api-data" followed by the name of the host and the UTC timestamp of the download time and the JSON file extension. The pattern is as follows:

```
api-data-fqdn-yyyy-mm-ddThh-mm-ss.json
```

For example:

```
api-data-fqdn-2.example.com-2017-06-15T09-00-32.json
```

**Disable Debug Mode**

When you are finished capturing and downloading debug files, you can disable debug mode.

- Click the *Disable debug mode* link in the notification message box.

The message box is dismissed and debug mode is disabled.

Or, use the Cloudera Navigator console menu:

- Select *user_name* > *Disable Debug Mode*.

**Set Debug Logging Levels**

The Cloudera Navigator Debug Mode allows you to configure the logging levels for individual methods in the application code. For example, to increase the logging detail for API calls, you would set the logging level for the `org.apache.cxf.interceptor` class to "DEBUG" level. If you are collecting debugging information for a specific issue, Cloudera support will provide the class name.

To set the logging level for a Cloudera Navigator application class:

1. *Log in to the Cloudera Navigator console.*
2. Add /debug to the Navigator URL:

[http://fqdn-a.example.com:7187/debug]

The debugging console appears.

3. Click Configure Logging.
4. In the Logger drop-down menu, select a Navigator class, such as org.apache.cxf.interceptor for API calls.
5. Set the logging level for the class.

   For example, for the API call logging, select DEBUG.

6. Click Submit.

   When changed in Debug Mode, logging levels will return to their default values after restarting the Navigator Metadata Server. To permanently change the logging levels, use Cloudera Manager to add an entry in the Navigator Metadata Server Logging Advanced Configuration Snippet. Create the entry as follows:

   ```
   log4j.logger.<classname>=<level>
   ```

   For example, to set the API logging level to "DEBUG", include the following string in the Advanced Configuration Snippet:

   ```
   log4j.logger.org.apache.cxf.interceptor=DEBUG
   ```

See Modifying Configuration Properties Using Cloudera Manager.

Applying Metadata to HDFS and Hive Entities using the API

Using the Navigator API and JSON formatted metadata definition files, entities can be assigned properties in bulk, prior to extraction.

Metadata Definition Files

You can add tags and properties to HDFS entities using metadata files. With metadata files, you can assign metadata to entities in bulk and create metadata before it is extracted. A metadata file is a JSON file with the following structure:

```json
{
  "name" : "name",
  "description" : "description",
  "properties" : {
    "key_name_1" : "value_1",
    "key_name_2" : "value_2"
  },
  "tags" : [ "tag_1" ]
}
```

To add metadata files to files and directories, create a metadata file with the extension `.navigator`, naming the files as follows:

- **File** - The path of the metadata file must be `.filename.navigator`. For example, to apply properties to the file `/user/test/file1.txt`, the metadata file path is `/user/test/.file1.txt.navigator`.
- **Directory** - The path of the metadata file must be `dirpath/.navigator`. For example, to apply properties to the directory `/user/`, the metadata path must be `/user/./navigator`.

The metadata file is applied to the entity metadata when the extractor runs.

Applying HDFS and Hive Metadata

The Navigator APIs can be used modify metadata of HDFS or Hive entities, such as databases, tables, and operations, before or after entity extraction.
• If an entity has been extracted when the API is called, the metadata is applied immediately.
• If the entity has not been extracted, you can preregister metadata, which is then applied once the entity is extracted.

The Navigator SDK includes examples of pre-registering entities. For example, see this example of [creating a Hive operation](https://example.com), which will allow you to see lineage for pre-registered entities.

Metadata is saved regardless of whether or not a matching entity is extracted. Cloudera Navigator does not perform any cleanup of unused metadata.

If you call the API before the entity is extracted, the custom metadata is stored with the entity's:

• Identity
• Source ID
• Metadata fields (name, description, tags, properties)
• Fields relevant to the identifier

Other fields (attributes) for the entity, such as Type, are not present.

To view all stored metadata, use the API to search for entities without an internal type, specifically the entities endpoint with the GET method:

```
http://fqdn-n.example.com:port/api/APIversion/entities/
```

where `fqdn-n.example.com` is the host running the Navigator Metadata Server role instance listening for HTTP connections at the specified `port` number (7187 is the default port number). `APIversion` is the running version of the API as indicated in the footer of the API documentation (available from the Help menu in the Navigator console) or by calling `http://fqdn-n.example.com:port/api/version`.

For example:

```
curl http://node1.example.com:7187/api/v13/entities/?query=-internalType:* -u username:password -X GET
```

User-defined metadata provided through the API overwrites existing metadata. For example, passing empty `name` and `description` fields with an empty array for `tags` and empty property dictionary with the API call removes the existing metadata. If you omit the tags or properties fields, the existing values remain unchanged. If you want to add a tag to a list of existing tags, you must include the existing tags in your update.

Modifying custom metadata using metadata files and the metadata API at the same time is not supported. You must use one or the other, because the two methods work differently.

Metadata specified in JSON files is merged with existing metadata, whereas the API overwrites metadata. Also, the updates provided by metadata files wait in a queue before being merged, but API changes are committed immediately. Some inconsistency can occur if a metadata file is merged when the API is in use.

Metadata is modified using either the **PUT** or **POST** method. Use the **PUT** method if the entity has been extracted, and the **POST** method to preregister metadata. Use the following syntax:

**PUT**

```
curl http://fqdn-n.example.com:port/api/APIversion/entities/identity 
-u username:password 
-X PUT 
-H "Content-Type: application/json" 
-d '{(properties)}'
```

where `identity` is an entity ID and `properties` are:

- name - Name metadata.
- description - Description metadata.
- tags - Tag metadata.
- properties - Custom metadata properties. The format is `{key: value}`.
- customProperties - Managed metadata properties. The format is `{namespace: {key: value}}`. If a property is assigned a value that does not conform to type constraints, an error is returned.

All existing naming rules apply, and if any value is invalid, the entire request is denied.

- POST

```bash
curl http://fqdn-n.example.com:port/api/APIversion/entities/ \
-u username:password \
-X POST \
-H "Content-Type: application/json" \
-d '{properties}'
```

where properties are:

- `sourceId` (required) - An existing source ID. After the first extraction, you can retrieve source IDs using the call:

```bash
curl http://fqdn-n.example.com:port/api/APIversion/entities/?query=type:SOURCE \
-u username:password -X GET
```

For example:

```json
[ ...
 {  "identity": "61cfef3d303d4284b7f5014b701f2c76d",
   "originalName": "source.listing",
   "originalDescription": null,
   "sourceId": "012437f9e9ebc23dc6e69679ac94d7fa2",
   "firstClassParentId": null,
   "parentPath": "/user/hdfs/.cm/distcp/2016-02-03_487",
   ...
   "properties": {
   "__cloudera_internal__hueLink": "http://fqdn-2.example.com:8888/filebrowser/#/user/hdfs/.cm/distcp/2016-02-03_487/source.listing",
   "fileSystemPath": "/user/hdfs/.cm/distcp/2016-02-03_487/source.listing",
   "type": "FILE",
   "permissions": "rw-r--r--",
   "owner": "hdfs",
   "group": "supergroup",
   "blockSize": 134217728,
   "mimeType": "application/octet-stream",
   "replication": 3,
   "userEntity": false,
   "deleted": false,
   "sourceType": "HDFS",
   "metaClassName": "fselement",
   "packageName": "nav",
   "internalType": "fselement"
   }, ...
]
```

If you have multiple services of a given type, you must specify the source ID that contains the entity you expect it to match.

- `parentPath` - The path of the parent entity, defined as:

  - HDFS file or directory - `fileSystemPath` of the parent directory. (Do not provide this field if the entity affected is the root directory.) Example `parentPath` for `/user/admin/input_dir:/user/admin`

If you add metadata to a directory, the metadata does not propagate to any files or folders in that directory.

- Hive database - If you are updating database metadata, do not specify this field.
- Hive table or view - The name of database containing the table or view. Example for a table in the default database: default.
- Hive column - `database name/table name/view name`. Example for a column in the sample_07 table: default/sample_07.
- `originalName` (required) - The name as defined by the source system.
- HDFS file or directory - Name of file or directory (ROOT if the entity is the root directory). Example `originalName` for /user/admin/input_dir: input_dir.
- Hive database, table, view, or column - The name of the database, table, view, or column.
  - Example for default database: default
  - Example for sample_07 table: sample_07
- `identity`
- Metadata fields (name, description, tags, properties)
- Fields relevant to the identifier

All existing naming rules apply, and if any value is invalid, the entire request is denied.

**API Usage Examples**

This section contains some examples.

**HDFS PUT Custom Metadata Example for /user/admin/input_dir Directory**

```
curl http://node1.example.com:7187/api/v13/entities/e461de8de38511a3ac6740dd7d51b8d0 \
-u username:password \
-X PUT \
-H "Content-Type: application/json"\n-d '{"name":"my_name", "description":"My description", "tags": ["tag1", "tag2"], "properties": {"property1": "value1", "property2": "value2"}}'
```

**HDFS POST Custom Metadata Example for /user/admin/input_dir Directory**

```
curl http://node1.example.com:7187/api/v13/entities/ \
-u username:password \
-X POST \
-H "Content-Type: application/json" \
-d '{"sourceId": "a09b0233cc58ff7d601eaa68673a20c6", "parentPath": "/user/admin", "originalName": "input_dir", "name": "my_name", "description": "My description", "tags": ["tag1", "tag2"], "properties": {"property1": "value1", "property2": "value2"}}'
```

**Hive POST Custom Metadata Example for total_emp Column**

```
curl http://node1.example.com:7187/api/v13/entities/ \
-u username:password \
-X POST \
-H "Content-Type: application/json" \
-d '{"sourceId": "4fbdadc6899638782fc8cb626176d7b", "parentPath": "default/sample_07", "originalName": "total_emp", "name": "my_name", "description": "My description", "tags": ["tag1", "tag2"], "properties": {"property1": "value1", "property2": "value2"}}'
```
HDFS PUT Managed Metadata Example

This example adds a property and a tag to the entity identified as "14", which happens to be the Hive `email_preferences` column in the `customers` sample table. The `Approved` property is a Boolean to indicate whether or not the metadata for this column was reviewed and approved. The `ToBeReviewed` tag marks the column temporarily so a data steward can easily find this column for review:

```bash
curl http://node1.example.com:7187/api/v13/entities/14 \
-u username:password \
-X PUT \
-H "Content-Type: application/json" \
-d '{"tags": ["ToBeReviewed"], \
  "customProperties": {"Operations": \n  {"Approved": false}}}'
```

The server responds:

```
{
  "originalName" : "email_preferences",
  "originalDescription" : null,
  "sourceId" : "7",
  "firstClassParentId" : "13",
  "parentPath" : "/default/customers",
  "deleteTime" : null,
  "extractorRunId" : "7##1",
  "customProperties" : {
    "Operations" : {
      "Approved" : true
    }
  },
  "name" : null,
  "description" : null,
  "tags" : [ "ToBeReviewed" ],
  "properties" : {
    "__cloudera_internal__hueLink" : 
    "https://node1.example.com:8889/hue/metastore/table/default/customers"
  },
  "technicalProperties" : null,
  "dataType" :
    "struct<email_format:string,frequency:string,categories:struct<promos:boolean,surveys:boolean>>",
  "type" : "FIELD",
  "sourceType" : "HIVE",
  "userEntity" : false,
  "metaClassName" : "hv_column",
  "deleted" : false,
  "packageName" : "nav",
  "identity" : "14",
  "internalType" : "hv_column"
}
```

GET Existing Metadata Examples

This section shows some examples of useful `GET /entity` calls:

To retrieve a specific entity:

Use the entity identifier that shows in the URL in the Navigator console. This example gets the metadata for the entity with identifier 21302.

```bash
curl https://node1.example.com:7187/api/v14/entities/21302 \
-X GET -b NavCookie
```

To retrieve entities that describe the sources Navigator extracts data from:

```bash
curl https://node1.example.com:7187/api/v14/entities/?query=type%3ASOURCE \
-b NavCookie -X GET
```
To use the source in a query, find its identity: In the output, find the name of the source, then collect the identity for that entity. In this example, the HIVE-1 source has the identity "6".

```json
{
  "originalName" : "HIVE-1",
  "originalDescription" : null,
  "sourceId" : null,
  "firstClassParentId" : null,
  "parentPath" : null,
  "deleteTime" : null,
  "extractorRunId" : null,
  "customProperties" : null,
  "name" : "HIVE-1",
  "description" : null,
  "tags" : null,
  "properties" : null,
  "technicalProperties" : null,
  "clusterName" : "Cluster 1",
  "sourceUrl" : "thrift://node2.example.com:9083",
  "sourceType" : "HIVE",
  "sourceExtractIteration" : 15,
  "sourceTemplate" : true,
  "hmsDbHost" : "node1.example.com",
  "hmsDbName" : "hive1",
  "hmsDbPort" : "3306",
  "hmsDbUser" : "hive1",
  "type" : "SOURCE",
  "userEntity" : false,
  "deleted" : null,
  "metaClassName" : "source",
  "packageName" : "nav",
  "identity" : "6",
  "internalType" : "source"
}
```

To retrieve all entities from a single source:

Run the previous call to determine the identity of the source, then use that value as the sourceId in the query. This example uses sourceId of 6.

```bash
curl https://node1.example.com:7187/api/v14/entities/?query=sourceId%3A6 \
-b NavCookie -X GET
```

To retrieve all entities marked with a specific tag:

Use quotes around the tag name. This example retrieves metadata for entities tagged with "sensitive".

```bash
curl https://node1.example.com:7187/api/v14/entities/?query=tags%3A%22sensitive%22 \
-b NavCookie -X GET
```

To retrieve all entities from a single source and marked with a tag:

This example includes a plus sign (+) before each of the query components to ensure they are treated as an AND relation. Note that this is a case where the query returns the correct results only with the ASCII codes.

```bash
curl https://node1.example.com:7187/api/v14/entities/?query=%2bsourceId%3A6%20%2bsourceId%3A6 \
-b cookie -X GET
```

To search for pre-registered entities:

This example shows how to get metadata for pre-registered entities, which do not include an internal type:

```bash
curl http://node1.example.com:7187/api/v13/entities/?query=-internalType:* \
-b cookie -X GET
```
Using the Cloudera Navigator SDK for Metadata Management

To facilitate working with metadata using the Cloudera Navigator APIs, Cloudera provides the Cloudera Navigator SDK. Cloudera Navigator SDK is a client library that provides functionality for extracting and enriching metadata with custom models, entities, and relationships. See GitHub cloudera/navigator-sdk for details.

Using the Purge APIs for Metadata Maintenance Tasks

Required Role: Cloudera Navigator Full Administrator

The volume of metadata maintained by Navigator Metadata Server can grow quickly and exceed the capacity of the Solr instance that processes the index and supports the search capability. For faster search and cleaner lineage tracing, use the purge feature to routinely delete unwanted metadata from the system.

Purging stale metadata is also recommended prior to upgrading an existing Cloudera Navigator instance. See Avoiding Out-of-Memory Errors During an Upgrade for details.

Note: Use the Cloudera Navigator console to purge deleted properties from managed metadata. See Deleting, Restoring, and Purging Managed Metadata Properties on page 34. As of Cloudera Navigator 2.11 (Cloudera Manager/CDH 5.12), the Cloudera Navigator console includes a configurable scheduling feature for Purge operations. See Defining Properties for Managed Metadata on page 31.

Purging Metadata for HDFS Entities, Hive and Impala Select Queries, and YARN, Sqoop, and Pig Operations

You can delete obsolete metadata using the purge method. Purge is a long-running task that requires exclusive access to the Solr instance and does not allow any concurrent activities, including extraction. For a list of what entities are affected by a purge task, see What Metadata is Purged? on page 81.

To purge metadata, do the following:

1. Back up the Navigator Metadata Server storage directory.
2. Invoke the maintenance/purge endpoint with the POST method with parameters from the following table:

   ```
   http://fqdn-n.example.com:port/api/APIversion/maintenance/purge?parameters
   ```

   where fqdn-n.example.com is the host running the Navigator Metadata Server role instance listening for HTTP connections at the specified port number (7187 is the default port number). APIversion is the running version of the API as indicated in the footer of the API documentation (available from the Help menu in the Navigator console) or by calling http://fqdn-n.example.com:port/api/version.

<table>
<thead>
<tr>
<th>Metadata</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
   | HDFS              | deleteTimeThresholdMinutes | After an HDFS entity is deleted, the amount of time that elapses before the entity can be purged.  
                      |                       | Default: 86400 minutes (60 days).                                            |
   |                   | runtimeCapMinutes     | Amount of time allowed for the HDFS purge process to run. When specified time is reached, the state is saved and the purge process stops. Run purge again to remove any remaining items held in state.  
                      |                       | Default: 720 minutes (12 hours). Set to 0 to effectively disable purge for HDFS files and directories (none will be purged).  
                      |                       | Default: 720 minutes (12 hours).                                             |
Set to true to enable purge for Hive and Impala SELECT queries,
and to enable YARN, Sqoop, and Pig operations older than the
staleQueryThresholdDays value to be purged.

Default: false
By default, does not purge Hive and Impala SELECT queries, nor are
YARN, Sqoop, and Pig operations purged.

Number of days at which Hive and Impala SELECT queries and YARN,
Sqoop, and Pig operations will be identified as stale, effectively
marking them 'ready for purge.' They will be purged automatically
within hours of the value being reached.

Default: 60 days. To disable marking entities as stale for the
foreseeable future, set the value to very large number, such as
36500.

For example, the following call purges the metadata of all deleted HDFS entities (elapsed minutes value set to 0):

```bash
$ curl -X POST -u admin:admin "http://node1.example.com:7187/api/v13/maintenance/purge?deleteTimeThresholdMinutes=0"
```

Purge tasks do not start until all currently running extraction tasks finish.

3. When all tasks have completed, click Continue to return to the Cloudera Navigator console.

Retrieving Purge Status

To view the status of the purge process, invoke the maintenance/running endpoint with the GET method:

```
http://fqdn-n.example.com:port/api/APIversion/maintenance/running
```

For example:

```bash
curl -X GET -u admin:admin "http://node1.example.com:7187/api/v13/maintenance/running"
```

A result would look similar to:

```json
[
  {
    "id" : 5,
    "type" : "PURGE",
    "startTime" : "2016-03-10T23:17:41.884Z",
    "endTime" : "1970-01-01T00:00:00.000Z",
    "status" : "IN_PROGRESS",
    "message" : "Purged 2661984 out of 4864714 directories. Averaging 1709 directories per minute."
    "username" : "admin",
    "stage" : "HDFS_DIRECTORIES",
    "stagePercent" : 54
  }
]
```

Retrieving Purge History

To view the purge history, invoke the maintenance/history endpoint with the GET method with parameters from
the following table:

```
http://fqdn-n.example.com:port/api/APIversion/maintenance/history?parameters
```
where *fqdn-n.example.com* is the host running the Navigator Metadata Server role instance listening for HTTP connections at the specified *port* number (7187 is the default port number). *APIversion* is the running version of the API as indicated in the footer of the API documentation (available from the Help menu in the Navigator console) or by calling [http://fqdn-n.example.com:port/api/version](http://fqdn-n.example.com:port/api/version).

### Table 2: History Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>First purge history entry to retrieve. Default: 0.</td>
</tr>
<tr>
<td>limit</td>
<td>Number of history entries to retrieve from the offset. Default: 100.</td>
</tr>
</tbody>
</table>

For example:

```
curl -X GET -u admin:admin
"http://node1.example.com:7187/api/v13/maintenance/history?offset=0&limit=100"
```

A result would look similar to:

```javascript
[{
  "id": 1,
  "type": "PURGE",
  "startTime": "2016-03-09T18:57:43.196Z",
  "endTime": "2016-03-09T18:58:33.337Z",
  "status": "SUCCESS",
  "username": "admin",
  "stagePercent": 0
},
{
  "id": 2,
  "type": "PURGE",
  "startTime": "2016-03-09T19:47:39.401Z",
  "endTime": "2016-03-09T19:47:40.841Z",
  "status": "SUCCESS",
  "username": "admin",
  "stagePercent": 0
},
{
  "id": 3,
  "type": "PURGE",
  "startTime": "2016-03-10T01:27:39.632Z",
  "endTime": "2016-03-10T01:27:46.809Z",
  "status": "SUCCESS",
  "username": "admin",
  "stagePercent": 0
},
{
  "id": 4,
  "type": "PURGE",
  "startTime": "2016-03-10T01:57:40.461Z",
  "endTime": "2016-03-10T01:57:41.174Z",
  "status": "SUCCESS",
  "username": "admin",
  "stagePercent": 0
},
{
  "id": 5,
  "type": "PURGE",
  "startTime": "2016-03-10T23:17:41.884Z",
  "endTime": "2016-03-10T23:18:06.802Z",
  "status": "SUCCESS",
  "username": "admin",
  "stagePercent": 0
}]
```
Cloudera Navigator APIs

Stopping a Purge Operation

To stop a purge operation that is running too long and blocking user access to Navigator, restart the Navigator Metadata Server. To complete the operation, rerun the purge command at a later time.
Cloudera Navigator Reference

This section provides the following reference information:

- **Lineage Diagram Icons** on page 159
- **Search Syntax and Properties** on page 162
- **Service Audit Events** on page 168
- **User Roles and Privileges Reference** on page 174

### Lineage Diagram Icons

**Entity Types and Icons Reference**

In lineage diagrams, entity types are represented by icons that vary depending upon the source system. The table below lists source system and shows the icons that can display in lineage diagrams.

Lineage diagrams are limited to 400 entities. After 400 entities, Cloudera Navigator lineage diagrams use the hidden icon to provide an entry point for exploring additional entities.

<table>
<thead>
<tr>
<th>Cluster</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Cluster Group" /></td>
<td>Cluster group</td>
</tr>
<tr>
<td><img src="image" alt="Cluster Instance" /></td>
<td>Cluster instance</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HDFS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="File" /></td>
<td>File</td>
</tr>
<tr>
<td><img src="image" alt="Directory" /></td>
<td>Directory</td>
</tr>
</tbody>
</table>

**Hive and Impala**

Hive entities include tables that result from Impala queries and Sqoop jobs.

<table>
<thead>
<tr>
<th>Hive and Impala</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Table" /></td>
<td>Table</td>
</tr>
<tr>
<td><img src="image" alt="Field" /></td>
<td>Field</td>
</tr>
<tr>
<td><img src="image" alt="Operation" /></td>
<td>Operation, sub-operation, execution</td>
</tr>
<tr>
<td><img src="image" alt="Operation" /></td>
<td>Impala operation, sub-operation, execution</td>
</tr>
</tbody>
</table>

**MapReduce and YARN**

<table>
<thead>
<tr>
<th>MapReduce and YARN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="MapReduce Operation" /></td>
<td>MapReduce operation and operation execution</td>
</tr>
<tr>
<td><img src="image" alt="YARN Operation" /></td>
<td>YARN operation and operation execution</td>
</tr>
</tbody>
</table>

**Oozie**

<table>
<thead>
<tr>
<th>Oozie</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Operation" /></td>
<td>Operation, operation execution</td>
</tr>
</tbody>
</table>
### Other Icons and Visual Elements in Lineage Diagrams

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🗒️</td>
<td><strong>Hidden</strong> entities. Used in lineage diagrams containing more than 400 entities. Click this icon to display the hidden entity details. See <a href="#">Exploring Hidden Entities in a Lineage Diagram</a> for more information.</td>
</tr>
</tbody>
</table>
| ...

### Types of Relations Shown in Lineage Diagrams

Lineage diagrams render relationships between entities using different line styles and colors. Arrows indicate data flow direction. Cloudera Navigator supports the following types of relations:
<table>
<thead>
<tr>
<th>Relation Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data flow</td>
<td>A relation between data and a processing activity, such as between a file and a MapReduce job or vice versa.</td>
</tr>
<tr>
<td></td>
<td>Sometime you may see a data flow relation without data assets. If that happens, it may be because they've been deleted. Turn on Deleted Entities in the lineage view to see the original data assets.</td>
</tr>
<tr>
<td>Parent-child</td>
<td>A parent-child relation. For example, the relation between a directory (parent) and a file (child).</td>
</tr>
<tr>
<td>Logical-physical</td>
<td>The relation between a logical entity and its physical entity. For example, between a Hive query and a MapReduce job.</td>
</tr>
<tr>
<td>Instance of</td>
<td>The relation between an operation execution and its operation. Instance of relations are not rendered in lineage diagrams.</td>
</tr>
<tr>
<td>Control flow</td>
<td>A relation in which a source entity controls the data flow of the target entity. For example, the relation between the columns of an insert clause and the where clause of a Hive query.</td>
</tr>
</tbody>
</table>

Lineage diagrams contain the line types shown in the table:

<table>
<thead>
<tr>
<th>Line Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A solid line depicts a data flow relationship, indicating that the columns appear (possibly transformed) in the output (when line has directional arrow) and logical-physical (when line does not have an arrow). For example, a solid line appears between the columns used in a select clause.</td>
</tr>
<tr>
<td></td>
<td>If you don't see the data assets involved, it may be because they've been deleted. Turn on Deleted Entities in the lineage view to see the original data assets.</td>
</tr>
<tr>
<td></td>
<td>A dashed line depicts a control flow relationship, indicating that the columns determine which rows flow to the output. For example, a dashed line appears between the columns used in an insert or select clause and the where clause of a Hive query. Control flow lines are hidden by default. See Filtering Lineage Diagrams on page 59.</td>
</tr>
<tr>
<td></td>
<td>A blue line depicts a link within the lineage diagram that has been selected.</td>
</tr>
<tr>
<td></td>
<td>A green line depicts a summary link that contains operations. When clicked, the green line turns blue denoting it has been selected, and the nested operations display in a selected link summary.</td>
</tr>
</tbody>
</table>

Selected Link Summary

Data Flow Summary

```
<table>
<thead>
<tr>
<th>sample_07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operations:</td>
</tr>
<tr>
<td>INSERT OVERWRITE TABLE sample_09</td>
</tr>
<tr>
<td>SELECT sample_07.co...</td>
</tr>
<tr>
<td>sample_09</td>
</tr>
</tbody>
</table>
```
Search Syntax and Properties

Cloudera Navigator search uses an embedded Solr engine that follows the syntax specified for the LuceneQParserPlugin.

Search Syntax

Search strings are constructed by specifying a default property value and one of the four types of key-value pairs as follows:

- **Technical metadata key-value pairs** - `key: value`
  - `key` is one of the properties listed in Searchable Properties Reference on page 163.
  - `value` is a single value or range of values specified as `[value1 TO value2]`. In a value, `*` is a wildcard. In property values, you must escape special characters `;`, `–`, `/`, and `*` with the backslash character (`\`), or enclose the property value in quotes.

  Technical metadata key-value pairs are read-only and cannot be modified.

- **Custom metadata key-value pairs** - `up_key:value`
  - `key` is a user-defined property.
  - `value` is a single value or range of values specified as `[value1 TO value2]`. In a value, `*` is a wildcard. In property values, you must escape special characters `;`, `–`, `/`, and `*` with the backslash character (`\`), or enclose the property value in quotes.

  Custom metadata key-value pairs can be modified.

- **Hive extended attribute key-value pairs** - `tp_key:value`
  - `key` is an extended attribute set on a Hive entity. The syntax of the attribute is specific to Hive.
  - `value` is a single value supported by the entity type.

  Hive extended attribute key-value pairs are read-only and cannot be modified.

- **Managed metadata key-value pairs** - `namespace.key:value`
  - `namespace` is the namespace containing the property. See Defining Properties for Managed Metadata on page 31.
  - `key` is the name of a managed metadata property.
  - `value` is a single value, a range of values specified as `[value1 TO value2]`, or a set of values separated by spaces. In a value, `*` is a wildcard. In property values, you must escape special characters `;`, `–`, `/`, and `*` with the backslash character (`\`), or enclose the property value in quotes.

  Only the values of managed metadata key-value pairs can be modified.

- **S3 key-value pairs** - `tp_key:value`
  - `key` is the name of user-defined metadata.
  - `value` is a single value.
  - Only file metadata is extracted; bucket and folder metadata is not extracted.

Constructing Compound Search Strings

To construct compound search strings, you can join multiple property-value pairs using the Lucene Query Parser Boolean operators:

- `+`, `-`
- OR, AND, NOT

In both syntax m use () to group multiple clauses into a single field and to form sub-queries.

When you filter Search results in the Cloudera Navigator console, the constructed search strings use the `+`, `-` syntax.
Example Search Strings

- Entities in the path /user/hive that have not been deleted -
  +"/user/hive") +(-deleted:true)
- Descriptions that start with the string "Banking" -
  description:Banking*
- Entities of type MapReduce or entities of type Hive -
  sourceType:mapreduce sourceType:hive or
  sourceType:mapreduce OR sourceType:hive
- Entities of type HDFS with size equal to or greater than 1024 MiB or entities of type Impala -
  +sourceType:hdfs +size:[1073741824 TO *) sourceType:impala
- Directories owned by hdfs in the path /user/hdfs/input -
  +owner:hdfs +type:directory
  +fileSystemPath:"/user/hdfs/input" or
  owner:hdfs AND type:directory AND
  fileSystemPath:"/user/hdfs/input"
- Job started between 20:00 to 21:00 UTC -
  started:[2013-10-21T20:00:00.000Z TO
  2013-10-21T21:00:00.000Z]
- Custom key-value -
  -project-customer1 - up_project:customer1
- Technical key-value - In Hive, specify table properties like this:

  ```sql
  ALTER TABLE table_name SET TBLPROPERTIES ('key1'='value1');
  ```

  To search for this property, specify tp_key1:value1.
- Managed key-value with multivalued property -
  MailAnnotation.emailTo:"dana@example.com"
  MailAnnotation.emailTo:"lee@example.com"

  Note: When viewing MapReduce jobs in the Cloudera Manager Activities page, the string that appears
  in a job Name column equates to the originalName property. To specify a MapReduce job name in
  a search, use the string (sourceType:mapreduce) and (originalName:jobName), where
  jobName is the value in the job Name column.

Searchable Properties Reference

The following sections describe the categories (types) of properties that can be searched.

Default Properties

The following properties can be searched by specifying a property value: type, fileSystemPath, inputs, jobId,
mapper, mimeType, name, originalName, outputs, owner, principal, reducer, and tags.

Common Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description</td>
<td>text</td>
<td>Description of the entity.</td>
</tr>
<tr>
<td>group</td>
<td>caseInsensitiveText</td>
<td>The group to which the owner of the entity belongs.</td>
</tr>
<tr>
<td>name</td>
<td>ngramedText</td>
<td>The overridden name of the entity. If the name has not been overridden, this value is empty. Names cannot contain spaces.</td>
</tr>
<tr>
<td>operationType</td>
<td>ngramedText</td>
<td>The type of an operation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Pig - SCRIPT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Sqoop - Table Export, Query Import</td>
</tr>
<tr>
<td>originalName</td>
<td>ngramedText</td>
<td>Entity name at extraction time.</td>
</tr>
<tr>
<td>originalDescription</td>
<td>text</td>
<td>Entity description at extraction.</td>
</tr>
<tr>
<td>owner</td>
<td>caseInsensitiveText</td>
<td>Entity owner.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>principal</td>
<td>caseInsensitiveText</td>
<td>For entities with type OPERATION_EXECUTION, the initiator of the entity.</td>
</tr>
<tr>
<td>properties</td>
<td>string</td>
<td>Set of key-value pairs that describe the entity.</td>
</tr>
<tr>
<td>tags</td>
<td>ngramedText</td>
<td>Set of tags that describe the entity.</td>
</tr>
<tr>
<td>type</td>
<td>tokenizedCaseInsensitiveText</td>
<td>Entity type. Source type of entity determines set of types available.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hdfs - directory, file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• hive - database, table, field, operation, operationExecution, subOperation, partition, view</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• impala - operation, operationExecution, subOperation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• mapreduce - operation, operationExecution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• oozie - operation, operationExecution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• pig - operation, operationExecution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• spark - operation, operationExecution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• sqoop - operation, operationExecution, subOperation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• yarn - operation, operationExecution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• cluster - clusterTemplate, clusterInstance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• s3 - file, s3bucket</td>
</tr>
<tr>
<td>userEntity</td>
<td>Boolean</td>
<td>Indicates whether an entity was added using the Cloudera Navigator SDK.</td>
</tr>
<tr>
<td>queryText</td>
<td>string</td>
<td>The text of a Hive, Impala, or Sqoop query.</td>
</tr>
<tr>
<td>clusterName</td>
<td>string</td>
<td>The name of the cluster in which the source is managed.</td>
</tr>
<tr>
<td>sourceId</td>
<td>string</td>
<td>The ID of the source type.</td>
</tr>
<tr>
<td>sourceType</td>
<td>caseInsensitiveText</td>
<td>The source type of the entity: hdfs, hive, impala, mapreduce, oozie, pig, spark, sqoop, or yarn.</td>
</tr>
<tr>
<td>sourceUrl</td>
<td>string</td>
<td>The URL of web application for a resource.</td>
</tr>
</tbody>
</table>

**timestamps**

Fields vary by source type:
- hdfs - created, lastAccessed, lastModified
- hive - created, lastModified
- impala, mapreduce, pig, spark, sqoop, and yarn - started, ended

<table>
<thead>
<tr>
<th>Date</th>
<th>Timestamps in the Solr Date Format. For example:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• lastAccessed: [ * TO NOW]</td>
</tr>
<tr>
<td></td>
<td>• created: [1976-03-06T23:59:59.999Z TO *]</td>
</tr>
<tr>
<td></td>
<td>• started: [1995-12-31T23:59:59.999Z TO</td>
</tr>
<tr>
<td></td>
<td>2007-03-06T00:00:00Z]</td>
</tr>
<tr>
<td></td>
<td>• ended: [NOW-1YEAR/DAY TO NOW/DAY+1DAY]</td>
</tr>
<tr>
<td></td>
<td>• created: [1976-03-06T23:59:59.999Z TO</td>
</tr>
<tr>
<td></td>
<td>1976-03-06T23:59:59.999Z+1YEAR]</td>
</tr>
<tr>
<td></td>
<td>• lastAccessed: [1976-03-06T23:59:59.999Z/YEAR TO</td>
</tr>
<tr>
<td></td>
<td>1976-03-06T23:59:59.999Z]</td>
</tr>
</tbody>
</table>

**Entity Types**
<table>
<thead>
<tr>
<th>Source</th>
<th>Entity Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>hdfs</td>
<td>DIRECTORY, FILE, DATASET, FIELD</td>
</tr>
<tr>
<td>hive</td>
<td>DATABASE, TABLE, FIELD, OPERATION, OPERATION_EXECUTION, SUB_OPERATION, PARTITION, RESOURCE, VIEW</td>
</tr>
<tr>
<td>impala</td>
<td>OPERATION, OPERATION_EXECUTION, SUB_OPERATION</td>
</tr>
<tr>
<td>mapreduce</td>
<td>OPERATION, OPERATION_EXECUTION</td>
</tr>
<tr>
<td>oozie</td>
<td>OPERATION, OPERATION_EXECUTION</td>
</tr>
<tr>
<td>pig</td>
<td>OPERATION, OPERATION_EXECUTION</td>
</tr>
<tr>
<td>spark</td>
<td>OPERATION, OPERATION_EXECUTION</td>
</tr>
<tr>
<td>sqoop</td>
<td>OPERATION, OPERATION_EXECUTION, SUB_OPERATION</td>
</tr>
<tr>
<td>yarn</td>
<td>OPERATION, OPERATION_EXECUTION, SUB_OPERATION</td>
</tr>
</tbody>
</table>

### Dataset Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>compressionType</td>
<td>tokenizedCaseInsensitiveText</td>
<td>The type of compression of a dataset file.</td>
</tr>
<tr>
<td>dataType</td>
<td>string</td>
<td>The data type: record.</td>
</tr>
<tr>
<td>datasetType</td>
<td>tokenizedCaseInsensitiveText</td>
<td>The type of the dataset: Kite.</td>
</tr>
<tr>
<td>fileFormat</td>
<td>tokenizedCaseInsensitiveText</td>
<td>The format of a dataset file: Avro or Parquet.</td>
</tr>
<tr>
<td>fullDataType</td>
<td>string</td>
<td>The full data type: record.</td>
</tr>
<tr>
<td>partitionType</td>
<td>string</td>
<td>The type of the partition.</td>
</tr>
<tr>
<td>schemaName</td>
<td>string</td>
<td>The name of the dataset schema.</td>
</tr>
<tr>
<td>schemaNameSpace</td>
<td>string</td>
<td>The namespace of the dataset schema.</td>
</tr>
</tbody>
</table>

### HDFS Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>blockSize</td>
<td>long</td>
<td>The block size of an HDFS file.</td>
</tr>
<tr>
<td>deleted</td>
<td>Boolean</td>
<td>Indicates whether the entity has been moved to the Trash folder.</td>
</tr>
<tr>
<td>deleteTime</td>
<td>date</td>
<td>The time the entity was moved to the Trash folder.</td>
</tr>
<tr>
<td>fileSystemPath</td>
<td>path</td>
<td>The path to the entity.</td>
</tr>
<tr>
<td>mimeType</td>
<td>ngramedText</td>
<td>The MIME type of an HDFS file.</td>
</tr>
<tr>
<td>parentPath</td>
<td>string</td>
<td>The path to the parent entity of a child entity. For example: parent path:/default/sample_07 for the table sample_07 from the Hive database default.</td>
</tr>
<tr>
<td>permissions</td>
<td>string</td>
<td>The UNIX access permissions of the entity.</td>
</tr>
<tr>
<td>replication</td>
<td>int</td>
<td>The number of copies of HDFS file blocks.</td>
</tr>
<tr>
<td>size</td>
<td>long</td>
<td>The exact size of the entity in bytes or a range of sizes. Range examples: size:[1000 TO <em>], size: [</em> TO 2000], and size:[* TO *] to find all fields with a size value.</td>
</tr>
</tbody>
</table>
## Hive Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dataType</td>
<td>ngramedText</td>
<td>The type of data stored in a field (column).</td>
</tr>
<tr>
<td>Table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>compressed</td>
<td>Boolean</td>
<td>Indicates whether a table is compressed.</td>
</tr>
<tr>
<td>serDeLibName</td>
<td>string</td>
<td>The name of the library containing the SerDe class.</td>
</tr>
<tr>
<td>serDeName</td>
<td>string</td>
<td>The fully qualified name of the SerDe class.</td>
</tr>
<tr>
<td>Partition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>partitionColNames</td>
<td>string</td>
<td>The table columns that define the partition.</td>
</tr>
<tr>
<td>partitionColValues</td>
<td>string</td>
<td>The table column values that define the partition.</td>
</tr>
<tr>
<td>technical_properties</td>
<td>string</td>
<td>Hive extended attributes.</td>
</tr>
<tr>
<td>clusteredByColNames</td>
<td>string</td>
<td>The column names that identify how table content is divided into buckets.</td>
</tr>
<tr>
<td>sortByColNames</td>
<td>string</td>
<td>The column names that identify how table content is sorted within a bucket.</td>
</tr>
</tbody>
</table>

## MapReduce and YARN Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inputRecursive</td>
<td>Boolean</td>
<td>Indicates whether files are searched recursively under the input directories, or only files directly under the input directories are considered.</td>
</tr>
<tr>
<td>jobId</td>
<td>ngramedText</td>
<td>The ID of the job. For a job spawned by Oozie, the workflow ID.</td>
</tr>
<tr>
<td>mapper</td>
<td>string</td>
<td>The fully qualified name of the mapper class.</td>
</tr>
<tr>
<td>outputKey</td>
<td>string</td>
<td>The fully qualified name of the class of the output key.</td>
</tr>
<tr>
<td>outputValue</td>
<td>string</td>
<td>The fully qualified name of the class of the output value.</td>
</tr>
<tr>
<td>reducer</td>
<td>string</td>
<td>The fully qualified name of the reducer class.</td>
</tr>
</tbody>
</table>

## Operation Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inputFormat</td>
<td>string</td>
<td>The fully qualified name of the class of the input format.</td>
</tr>
<tr>
<td>outputFormat</td>
<td>string</td>
<td>The fully qualified name of the class of the output format.</td>
</tr>
<tr>
<td>Operation Execution</td>
<td></td>
<td></td>
</tr>
<tr>
<td>inputs</td>
<td>string</td>
<td>The name of the entity input to an operation execution. For entities of resource type mapreduce, yarn, and spark, it is usually a directory. For entities of resource type hive, it is usually a table.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>outputs</td>
<td>string</td>
<td>The name of the entity output from an operation execution. For entities of resource type <code>mapreduce</code>, <code>yarn</code>, and <code>spark</code>, it is usually a directory. For entities of resource type <code>hive</code>, it is usually a table.</td>
</tr>
<tr>
<td>engineType</td>
<td>string</td>
<td>The type of the engine used for an operation: MR or Spark.</td>
</tr>
</tbody>
</table>

### Oozie Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>string</td>
<td>The status of the Oozie workflow: RUNNING, SUCCEEDED, or FAILED.</td>
</tr>
</tbody>
</table>

### Pig Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scriptId</td>
<td>string</td>
<td>The ID of the Pig script.</td>
</tr>
</tbody>
</table>

### S3 Properties

#### Object Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>string</td>
<td>The geographic region in which the bucket is stored</td>
</tr>
<tr>
<td>bucketName</td>
<td>string</td>
<td>The name of the bucket in which the object is stored</td>
</tr>
<tr>
<td>fileSystemPath</td>
<td>path</td>
<td>The key of the S3 object.</td>
</tr>
<tr>
<td>size</td>
<td>long</td>
<td>Object size in bytes.</td>
</tr>
<tr>
<td>lastModified</td>
<td>date</td>
<td>Object creation date or the last modified date, whichever is the latest.</td>
</tr>
<tr>
<td>etag</td>
<td>string</td>
<td>A hash of the object. The ETag reflects changes only to the contents of an object, not its metadata. The ETag may or may not be an MD5 digest of the object data.</td>
</tr>
<tr>
<td>storageClass</td>
<td>string</td>
<td>Storage class used for storing the object.</td>
</tr>
<tr>
<td>owner</td>
<td>string</td>
<td>Owner of the object.</td>
</tr>
<tr>
<td>sequencer</td>
<td>string</td>
<td>Latest S3 event notification sequencer. Used to order events.</td>
</tr>
<tr>
<td>parentPath</td>
<td>string</td>
<td>Parent of the S3 object.</td>
</tr>
<tr>
<td>technicalProperties</td>
<td>key-value pairs</td>
<td>Custom metadata for each S3 object.</td>
</tr>
</tbody>
</table>

#### Bucket Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>string</td>
<td>Region for the bucket.</td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td>Date the bucket was created.</td>
</tr>
<tr>
<td>owner</td>
<td>string</td>
<td>Owner of the bucket.</td>
</tr>
</tbody>
</table>
### Sqoop Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbURL</td>
<td>string</td>
<td>The URL of the database from or to which the data was imported or exported.</td>
</tr>
<tr>
<td>dbTable</td>
<td>string</td>
<td>The table from or to which the data was imported or exported.</td>
</tr>
<tr>
<td>dbUser</td>
<td>string</td>
<td>The database user.</td>
</tr>
<tr>
<td>dbWhere</td>
<td>string</td>
<td>A where clause that identifies which rows were imported.</td>
</tr>
<tr>
<td>dbColumnExpression</td>
<td>string</td>
<td>An expression that identifies which columns were imported.</td>
</tr>
</tbody>
</table>

### Service Audit Events

Service audit events are the events generated by a given service running on the cluster. Users with the appropriate permissions (Auditing Viewer, Full Administrator) can view audit events in the Cloudera Navigator console or by using the APIs. Audit events can include the fields listed in the tables below. The field names differ between the Navigator API and the events as they appear when streaming to Kafka or syslog.

The Cloudera Navigator console Audits includes events collected by Cloudera Manager: service lifecycle events (activate, create, delete, deploy, download, install, start, stop, update, upgrade, and so on) and user security-related events (add and delete user, login failed and succeeded). See Lifecycle and Security Auditing for more details on Cloudera Manager audit events.

<table>
<thead>
<tr>
<th>Display Name</th>
<th>Field in API</th>
<th>Field in Streaming</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional Info</td>
<td>additional_info</td>
<td>additionalInfo</td>
<td>JSON text that contains more details about an operation performed on entities in Navigator Metadata Server.</td>
</tr>
<tr>
<td>Allowed</td>
<td>allowed</td>
<td>allowed</td>
<td>Indicates whether the request to perform an operation failed or succeeded. A failure occurs if the user is not authorized to perform the action.</td>
</tr>
<tr>
<td>Collection Name</td>
<td>collection_name</td>
<td>collectionName</td>
<td>The name of the affected Solr collection.</td>
</tr>
<tr>
<td>Database Name</td>
<td>database_name</td>
<td>db databaseName (Sentry)</td>
<td>For Sentry, Hive, and Impala, the name of the database on which the operation was performed.</td>
</tr>
<tr>
<td>Delegation Token ID</td>
<td>delegation_token_id</td>
<td>delegationTokenId</td>
<td>Delegation token identifier generated by HDFS NameNode that is then used by clients when submitting a job to JobTracker.</td>
</tr>
<tr>
<td>Destination</td>
<td>dest</td>
<td>dst</td>
<td>Path of the final location of an HDFS file in a rename or move operation.</td>
</tr>
<tr>
<td>Entity ID</td>
<td>entity_id</td>
<td>entityId</td>
<td>Identifier of a Navigator Metadata Server entity. The ID can be retrieved using the Navigator Metadata Server API.</td>
</tr>
<tr>
<td>Event Time</td>
<td>timestamp</td>
<td>time</td>
<td>Date and time an action was performed. The Navigator Audit Server stores the timestamp in the timezone of the Navigator Audit Server. The Cloudera Navigator console displays the timestamp converted to the local timezone. Exported audit events contain the stored timestamp.</td>
</tr>
<tr>
<td>Family</td>
<td>family</td>
<td>family</td>
<td>HBase column family.</td>
</tr>
<tr>
<td>Impersonator</td>
<td>impersonator</td>
<td>impersonator</td>
<td>Name of user (service) that invokes an action on behalf of another user (service). Impersonator field always displays</td>
</tr>
</tbody>
</table>

---

168 | Cloudera Navigator Data Management
<table>
<thead>
<tr>
<th>Display Name</th>
<th>Field in API</th>
<th>Field in Streaming</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP Address</td>
<td>ipAddress</td>
<td>ip</td>
<td>The IP address of the host where an action occurred.</td>
</tr>
<tr>
<td>Object Type</td>
<td>object_type</td>
<td>objType</td>
<td>For Sentry, Hive, and Impala, the type of the object (TABLE, VIEW, DATABASE) on which operation was performed.</td>
</tr>
<tr>
<td>Operation</td>
<td>command</td>
<td>op</td>
<td>Commands executed by component. See Operations by Component on page 170 for details. For Cloudera Navigator operations, see Navigator Metadata Server Sub Operations on page 172.</td>
</tr>
<tr>
<td>Operation Params</td>
<td>operation_params</td>
<td>operationParams</td>
<td>Solr query or update parameters used when performing the action.</td>
</tr>
<tr>
<td>Operation Text</td>
<td>operation_text</td>
<td>opText</td>
<td>For Sentry, Hive, and Impala, the SQL query that was executed by user. For Hue, the user or group that was added, edited, or deleted.</td>
</tr>
<tr>
<td>Permissions</td>
<td>permissions</td>
<td>perms</td>
<td>HDFS permission of the file or directory on which the HDFS operation was performed.</td>
</tr>
<tr>
<td>Privilege</td>
<td>privilege</td>
<td>privilege</td>
<td>Privilege needed to perform an Impala operation.</td>
</tr>
<tr>
<td>Qualifier</td>
<td>qualifier</td>
<td>qualifier</td>
<td>HBase column qualifier.</td>
</tr>
<tr>
<td>Query ID</td>
<td>query_id</td>
<td>—</td>
<td>The query ID for an Impala operation. (Internal use only)</td>
</tr>
<tr>
<td>Resource</td>
<td>resource</td>
<td>—</td>
<td>A service-dependent combination of multiple fields generated during fetch. This field is not supported for filtering as it is not persisted.</td>
</tr>
<tr>
<td>Resource Path</td>
<td>resource_path</td>
<td>path</td>
<td>HDFS URL of Hive objects (TABLE, VIEW, DATABASE, and so on). Used for HDFS, Sentry.</td>
</tr>
<tr>
<td>Service Name</td>
<td>service</td>
<td>service</td>
<td>The name of the service that performed the action.</td>
</tr>
<tr>
<td>Session ID</td>
<td>session_id</td>
<td>—</td>
<td>Impala session ID. (Internal use only)</td>
</tr>
<tr>
<td>Solr Version</td>
<td>solr_version</td>
<td>solrVersion</td>
<td>Solr version number.</td>
</tr>
<tr>
<td>Source</td>
<td>src</td>
<td>src</td>
<td>Path of the HDFS file or directory present in an HDFS operation.</td>
</tr>
<tr>
<td>Status</td>
<td>status</td>
<td>status</td>
<td>Status of an Impala operation providing more information on success or failure.</td>
</tr>
<tr>
<td>Stored Object Name</td>
<td>stored_object_name</td>
<td>name</td>
<td>Name of a policy, saved search, or audit report in Navigator Metadata Server.</td>
</tr>
<tr>
<td>Sub Operation</td>
<td>sub_operation</td>
<td>subOperation</td>
<td>Operations performed by Navigator Metadata Server are identified by subsystem (authorization, auditing, for example) and by sub-operation within that subsystem. See Navigator Metadata Server Sub Operations on page 172 for details.</td>
</tr>
</tbody>
</table>
### Operations by Component

The Operation field of an audit event includes the actions taken (commands run) on the component. Operations for Cloudera Navigator (and sub-operations) are listed [Navigator Metadata Server Sub Operations](#) on page 172.

**Note:**

Cloudera Navigator does not capture audit events for queries that are run on Spark or HiveServer1/Hive CLI. If you want to use Cloudera Navigator to capture auditing for Hive operations, upgrade to HiveServer2 if you have not done so already.

<table>
<thead>
<tr>
<th>Component</th>
<th>Action taken</th>
</tr>
</thead>
<tbody>
<tr>
<td>HBase</td>
<td>addColumn, append, assign, balance, balanceSwitch, checkAndDelete, checkAndPut, compact, compactSelection, createTable, delete, deleteColumn, deleteTable, disableTable, enableTable, exists, flush, get, getClosestRowBefore, grant, increment, incrementColumnValue, modifyColumn, modifyTable, move, put, revoke, scannerOpen, shutdown, split, stopMaster, unassign</td>
</tr>
<tr>
<td>HDFS</td>
<td>append, concat, create, createSymlink, delete, fsck, getfacl*, getFileInfo, listSnapshottableDirectory, listStatus, mkdirs, open, rename, setfacl*, setOwner, setPermission, setReplication, setTimes</td>
</tr>
<tr>
<td>HiveServer2/Beeline</td>
<td>ALTER_PARTITION_MERGE, ALTER_TABLE_MERGE, ALTERDATABASE, ALTERDATABASE_SET_OWNER, ALTERINDEX_PROPS, ALTERINDEX_REBUILD, ALTERPARTITION_FILEFORMAT, ALTERPARTITION_LOCATION, ALTERPARTITION_PROTECTMODE, ALTERPARTITION_SERDEPROPERTIES, ALTERPARTITIONIALIZER, ALTERTABLE_ADDCOLUMNS, ALTERTABLE_ADDPARTITIONS, ALTERTABLE_ARCHIVE, ALTERTABLE_CLUSTER_SORT, ALTERTABLE_DROPPARTITIONS, ALTERTABLE_FILEFORMAT, ALTERTABLE_LOCATION, ALTERTABLE_PROPERTIES, ALTERTABLE_PROTECTMODE, ALTERTABLE_RENAME, ALTERTABLE_RENAMECOLUMN, ALTERTABLE_RENAMEPARTITION, ALTERTABLE_REPLACECOLUMNS, ALTERTABLE_SERDEPROPERTIES, ALTERTABLE_SERIALIZE, ALTERN_TABLE_SET_OWNER, ALTERTABLE TOUCH, ALTERTABLE UNARCHIVE, ALTERVIEW_PROPERTIES, CREATEDATABASE, CREATEFUNCTION, CREATEINDEX, CREATEROLE, CREATETABLE AS SELECT, CREATETABLE, CREATEVIEW, DESCDATABASE, DESCFUNCTION, DESCTABLE, DROPDATABASE, DROPFUNCTION, DROPINDEX, DROPROLE, DROPTABLE, DROPVIEW, EXPLAIN, EXPORT, GRANT_PRIVILEGE, GRANT_ROLE, IMPORT, LOAD, LOCKTABLE, MSCK, QUERY, REVOKE_PRIVILEGE, REVOKE_ROLE, SHOW GRANT, SHOW ROLE GRANT, SHOW TABLESTATUS, SHOW_TBLPROPERTIES, SHOWDATABASES, SHOWFUNCTIONS, SHOWINDEXES, SHOWLOCKS, SHOWPARTITIONS, SHOWTABLES, SWITCHDATABASE, UNLOCKTABLE</td>
</tr>
</tbody>
</table>

See also [Data Manipulation Language statements](#)
**Component** | **Action taken** | **Hive and DML Statements Support**
--- | --- | ---
**Impala** | ALTER DATABASE SET OWNER, ALTER TABLE SET OWNER, ALTER VIEW SET OWNER, CREATE ROLE, DELETE, DROP ROLE, GRANT privilege, GRANT ROLE, INSERT, Query, REVOKE privilege, REVOKE ROLE, SHOW GRANT ROLE, SHOW ROLE GRANT, UPDATE | *See HDFS Audit Logging for ACL Operations on page 171.*
**Sentry** | ADD_ROLE_TO_GROUP, CREATE_ROLE, DELETE_ROLE_FROM_GROUP, DROP_ROLE, GRANT PRIVILEGE, REVOKE PRIVILEGE |  
**Sor** | add, ALTER DATABASE SET OWNER, ALTER TABLE SET OWNER, ALTER DATABASE SET OWNER, ALTER TABLE SET OWNER, ALTER VIEW SET OWNER, commit, CREATE, CREATEALIAS, CREATESHARD, DELETE, DELETEALIAS, deleteById, deleteByQuery, DELETESHARD, finish, LIST, LOAD_ON_STARTUP, LOAD, MERGEINDEXES, PERSIST, PREPRECOVERY, query, RELOAD, RENAME, REQUESTAPPLYUPDATES, REQUESTRECOVERY, REQUESTSYNCSHARD, rollback, SPLIT, SPLITSHARD, STATUS, SWAP, SYNCSHARD, TRANSIENT, UNLOAD |  

**HDFS Audit Logging for ACL Operations**

HDFS audit logging for ACL operations has some variation based on the command options.

<table>
<thead>
<tr>
<th>Command</th>
<th>Option</th>
<th>Audit Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>getfacl</td>
<td>—</td>
<td>getAclStatus</td>
</tr>
<tr>
<td>setfacl</td>
<td>--b</td>
<td>removeAcl</td>
</tr>
<tr>
<td>setfacl</td>
<td>--k</td>
<td>removeDefaultAcl</td>
</tr>
<tr>
<td>setfacl</td>
<td>--m</td>
<td>modifyAclEntries</td>
</tr>
<tr>
<td>setfacl</td>
<td>--x</td>
<td>removeAclEntries</td>
</tr>
<tr>
<td>setfacl</td>
<td>--set</td>
<td>setAcl</td>
</tr>
</tbody>
</table>

There is a difference in audit logging behavior based on how the ACL operations are run:

- Over FileSystem ACL APIs, all setfacl and getfacl operations produce audit log events.
- Over FsShell (that is, `hadoop fs` or `hdfs dfs` command lines):
  - All setfacl operations produce audit log events.
  - getfacl operations produce audit log events only if the file has ACLs set.

That is, setfacl operations always produce audit log events and getfacl operations always produce audit log events when ACLs are set.

**Hive DDL and DML Statements Support**

The table below lists Hive DDL and DML statements and whether Cloudera Navigator supports the operation with metadata and lineage extraction. This list applies to Cloudera Navigator 2.12 (Cloudera Manager 5.13) and later releases.

<table>
<thead>
<tr>
<th>Hive operations</th>
<th>Cloudera Navigator Support</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>✘</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Alter Table/Partition/Column</td>
<td>✔</td>
<td>Known Issue. ALTER TABLE RENAME TO does not create query entity. ALTER TABLE CHANGE column name does not create query operation entity.</td>
</tr>
<tr>
<td>Create Table</td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>
## Hive operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cloudera Navigator Support</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create/Drop Macro</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Create/Drop/Alter Index</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Create/Drop/Alter View</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Create/Drop/Alter/Use Database</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Create/Drop/Grant/Revoke Roles and Privileges</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Create/Drop/Reload Function</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>DELETE</td>
<td>✗</td>
<td>Requires ACID support. <strong>Hive ACID not supported.</strong></td>
</tr>
<tr>
<td>Describe</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>Drop/Truncate Table</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>EXPORT</td>
<td>✗</td>
<td>Known Issue. External tables can be exported to HDFS but Cloudera Navigator does not create a query entity for the EXPORT.</td>
</tr>
<tr>
<td>IMPORT</td>
<td>✗</td>
<td>Known Issue. External tables can be imported from HDFS but Cloudera Navigator does not create a query entity for the IMPORT.</td>
</tr>
<tr>
<td>INSERT data into Hive Tables from queries</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>INSERT data into the file system from queries</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>INSERT values into tables from SQL</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>LOAD</td>
<td>✗</td>
<td>Known Issue. LOADing a CSV from HDFS into an existing Hive table does not generate lineage.</td>
</tr>
<tr>
<td>MERGE</td>
<td>✗</td>
<td>Requires ACID support. <strong>Hive ACID not supported.</strong></td>
</tr>
<tr>
<td>MSCK REPAIR</td>
<td>✗</td>
<td>Tables track their respective partitions. Queries to create or repair partitions using MSCK are not captured as query entities.</td>
</tr>
<tr>
<td>Show</td>
<td>✗</td>
<td>Operation does not generate data flow lineage.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>✗</td>
<td>Requires ACID support. <strong>Hive ACID not supported.</strong></td>
</tr>
</tbody>
</table>

### Navigator Metadata Server Sub Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>Sub Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>auditReport</td>
<td>createAuditReport, deleteAuditReport, fetchAllReports, updateAuditReport</td>
</tr>
<tr>
<td>authorization</td>
<td>deleteGroup, fetchGroup, fetchRoles, searchGroup, updateRoles</td>
</tr>
<tr>
<td>metadata</td>
<td>fetchAllMetadata, fetchMetadata, updateMetadata</td>
</tr>
<tr>
<td>policy</td>
<td>createPolicy, deletePolicy, deletePolicySchedule, fetchAllPolicies, fetchPolicySchedule, updatePolicy, updatePolicySchedule</td>
</tr>
</tbody>
</table>
Service Metadata Entity Types

Metadata entities are the data assets stored in and operations generated by services running on the cluster. Users with the appropriate permissions (Metadata and Lineage Viewer, Full Administrator) can view metadata entities in the Cloudera Navigator console or by using the APIs.

The Cloudera Navigator console includes entities extracted from services that are enabled for integration with Navigator Metadata Service.

Navigator creates the following entities from metadata extracted from these services:

<table>
<thead>
<tr>
<th>Service</th>
<th>Navigator Entity Types Created from Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>directory, file</td>
</tr>
<tr>
<td>HiveServer2</td>
<td>database, table, field, operation, operation_execution, sub_operation, partition, view</td>
</tr>
<tr>
<td>Impala</td>
<td>operation, operation_execution, sub_operation</td>
</tr>
<tr>
<td>Mapreduce (v1 and v2)</td>
<td>operation, operation_execution</td>
</tr>
<tr>
<td>Oozie</td>
<td>operation, operation_execution</td>
</tr>
<tr>
<td>Pig</td>
<td>operation, operation_execution</td>
</tr>
<tr>
<td>Spark (v1 and v2)</td>
<td>operation, operation_execution</td>
</tr>
<tr>
<td>Sqoop (v1)</td>
<td>operation, operation_execution, sub_operation</td>
</tr>
<tr>
<td>YARN</td>
<td>operation, operation_execution</td>
</tr>
<tr>
<td>Cluster</td>
<td>cluster_template, cluster_instance</td>
</tr>
<tr>
<td>S3</td>
<td>file, s3bucket</td>
</tr>
</tbody>
</table>

Hive Operations and Cloudera Navigator Support Matrix

The table below lists Hive DDL and DML statements and whether Cloudera Navigator supports the operation with metadata and lineage extraction.

<table>
<thead>
<tr>
<th>Hive operations</th>
<th>Cloudera Navigator Metadata Support</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>☒</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Alter Table/Partition/Column</td>
<td>☑</td>
<td>ALTER TABLE RENAME TO does not create query entity. ALTER TABLE CHANGE column name does not create query entity.</td>
</tr>
<tr>
<td>Create Table</td>
<td>☑</td>
<td></td>
</tr>
<tr>
<td>Create/Drop Macro</td>
<td>☒</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Create/Drop/Alter Index</td>
<td>☒</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Create/Drop/Alter View</td>
<td>☒</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Hive operations</td>
<td>Cloudera Navigator Metadata Support</td>
<td>Comment</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Create/Drop/Alter/Use Database</td>
<td>☐</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Create/Drop/Grant/Revoke Roles and Privileges</td>
<td>☐</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Create/Drop/Reload Function</td>
<td>☐</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>DELETE</td>
<td>☐</td>
<td>Requires ACID support. <a href="#">Hive ACID not supported</a>.</td>
</tr>
<tr>
<td>Describe</td>
<td>☐</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>Drop/Truncate Table</td>
<td>☐</td>
<td></td>
</tr>
<tr>
<td>EXPORT</td>
<td>☐</td>
<td>Known Issue. External tables can be exported to HDFS but Cloudera Navigator does not create a query entity for the EXPORT.</td>
</tr>
<tr>
<td>IMPORT</td>
<td>☐</td>
<td>Known Issue. External tables can be imported from HDFS but Cloudera Navigator does not create a query entity for the IMPORT.</td>
</tr>
<tr>
<td>INSERT data into Hive Tables from queries</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>INSERT data into the filesystem from queries</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>INSERT values into tables from SQL</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>LOAD</td>
<td>☐</td>
<td>Known Issue. LOADING a CSV from HDFS into an existing Hive table does not generate lineage.</td>
</tr>
<tr>
<td>MERGE</td>
<td>☐</td>
<td>Requires ACID support. <a href="#">Hive ACID not supported</a>.</td>
</tr>
<tr>
<td>MSCK REPAIR</td>
<td>☐</td>
<td>Tables track their respective partitions. Queries to create or repair partitions using <a href="#">MSCK</a> are not captured as query entities.</td>
</tr>
<tr>
<td>Show</td>
<td>☐</td>
<td>Operation does not generate data flow.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>☐</td>
<td>Requires ACID support. <a href="#">Hive ACID not supported</a>.</td>
</tr>
</tbody>
</table>

Cloudera Navigator extractor for Hive does not support all Hive statements, specifically these:

- Table-generating functions
- Lateral views
- Transform clauses
- Regular expressions (regex) in SELECT clause

Hive queries that include any of the above will prevent lineage diagrams from completing successfully.

### User Roles and Privileges Reference

Cloudera Navigator lets authenticated users—those who have successfully logged in to Cloudera Navigator console—access features and functionality according to specifically assigned user roles. The Cloudera Navigator user
role scheme requires the use of an external LDAP-compliant identity-directory service, such as OpenLDAP or Microsoft Server Active Directory. See Cloudera Navigator Authentication Overview for details.

After users have been assigned to various groups in the external LDAP-compliant identity-directory service, Cloudera Navigator administrators can grant user roles to various groups as needed. See Administering Navigator User Roles on page 78 for details.

Information about user roles and the specific privileges to Cloudera Navigator is contained in the high-level summary and detailed matrix below.

### Cloudera Navigator User Roles

Cloudera Navigator provides the following user roles for organizing and applying user permissions:

- **Auditing Viewer**—View audit events, view audit analytics, and create audit reports.
- **Custom Business Metadata Editor**—Search for entities, view metadata, view lineage, view metadata analytics, edit user-defined properties and tags.
- **Managed and Custom Business Metadata Editor**—Search for entities, view metadata, view lineage, edit user-defined properties, edit managed properties, and define managed metadata models.
- **Metadata & Lineage Viewer**—Search for entities, view metadata, view lineage, and view metadata analytics such as the Data Stewardship Dashboard and the Data Explorer.
- **Policy Editor**—View, create, update, and delete metadata policies.
- **Policy Viewer**—View policies that can be applied to metadata. Note that this role needs to be combined with other roles to give view access to Navigator metadata.
- **User Administrator**—Assign user roles to groups. Requires the group (and the user accounts it contains) to exist in an LDAP-compliant directory service. See Configuring Navigator User Roles and Authentication and Cloudera Navigator for details.
- **Full Administrator**—All Navigator permissions.

Cloudera Manager provides the following user roles for managing Cloudera Manager and Cloudera Navigator.

- **Full Administrator**—Complete administrative and management access, including assigning user roles to groups. Required to install Cloudera Navigator. Users with this role are automatically assigned the Navigator Full Administrator role.
- **Navigator Administrator**—View data in Cloudera Manager, view service and monitoring information, administer Cloudera Navigator, view audit events. Users with this role are automatically assigned the Navigator Full Administrator role.

### Cloudera Navigator User Role Details

The Cloudera Manager user roles **Full Administrator** and **Navigator Administrator** have the privileges listed for Full Administrator in the table below.

<table>
<thead>
<tr>
<th>Cloudera Navigator User Role</th>
<th>Auditing Viewer</th>
<th>Custom Business Metadata Editor</th>
<th>Managed and Custom Business Metadata Editor</th>
<th>Metadata &amp; Lineage Viewer</th>
<th>Policy Editor</th>
<th>Policy Viewer</th>
<th>User Administrator</th>
<th>Full Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role-to-group mapping</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Define metadata models</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Edit entity name or description</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
</tbody>
</table>
Default HFDS Audit Filters

The following JSON includes the recommended default HDFS audit filters. If the audit filters here do not match what you see in the Cloudera Manager HDFS configuration, it is likely that these defaults were not applied when Cloudera Manager was upgraded to the latest release because they would overwrite customizations made in previous releases.

See Default and Recommended Audit Filters on page 97 for explanations of the filters and how to customize them.

```json
{
    "comment": "The default HDFS audit event filter accepts all denied access, delete, and rename events, and discards events that affects files in any of the staging directories (Hive, Spark, Impala), events that affect files in /tmp directory, events that affect files in Cloudera Hive Canary directory, "events generated by the internal Cloudera and Hadoop users (cloudera-scm, dr.who, hbase, hive, impala, mapred, solr, and spark), and 'ls' actions", "performed by the hdfs user."
}
```
{
  "defaultAction": "accept",
  "rules": [
    {
      "action": "accept",
      "fields": [
        {
          "name": "allowed",
          "match": "(?i:allow)"
        }
      ]
    },
    {
      "action": "discard",
      "fields": [
        {
          "name": "src",
          "match": "(.*/\.hive-staging($|/.*))?|.*/\.staging($|/.*))?|.*/\.sparkStaging($|/.*))?|.*/_impala_insert_staging($|/.*))?|/user/his ... astore_canary(?:/.*)?|/user/hue/\.Trash/Current/user/hue/\.cloudera_manager_hive_metastore_canary(?:/.*)?|/tmp(?:/.*)?"
        }
      ]
    },
    {
      "action": "accept",
      "fields": [
        {
          "name": "operation",
          "match": "delete|rename.*"
        }
      ]
    },
    {
      "action": "discard",
      "fields": [
        {
          "name": "username",
          "match": "(?i:cloudera-scm|dr.who|hbase|hive|impala|mapred|solr|spark)(?:/.+)?"
        }
      ]
    },
    {
      "action": "discard",
      "fields": [
        {
          "name": "operation",
          "match": "listStatus|listCachePools|listCacheDirectives|getfileinfo"
        }
      ]
    },
    {
      "action": "accept",
      "fields": [
        {
          "name": "operation",
          "match": "getfileinfo"
        }
      ]
    }
  ]
}
Troubleshooting Navigator Data Management

This page contains troubleshooting tips and workarounds for various issues that can arise.

Cloudera Navigator-Cloudera Altus

No metadata or lineage collected from an Altus deployed cluster.

**Symptom:** No metadata collected from a Cloudera Altus instantiated cluster. Although both the Cloudera Altus environment has been setup to use a given Amazon S3 bucket and the Cloudera Navigator instance has been configured to read from that same S3 bucket, no metadata is collected.

**Possible cause:** Permissions on the Amazon S3 bucket have not been applied correctly.

**Steps to resolve:** To determine the root cause of this issue:

- Check that permissions have been set properly on the Amazon S3 bucket.
- Verify that the Amazon S3 bucket name has not been changed.

To determine if the cause of the issue is misconfiguration, look at the Telemetry Publisher logs. A failure message in the log such as the following means that the Amazon S3 bucket has either not been properly identified, or that the correct permissions have not been set:

```
...Metadata export failed com.amazonaws.services.s3.model.AmazonS3Exception: The specified bucket does not exist
(Service: Amazon S3; Status Code: 404; Error Code: NoSuchBucket; Request ID: D28543E2494BD521), S3 Extended Request ID:
...
```

The 404 status code and "NoSuchBucket" in the error message indicate the Telemetry Publisher was unable to write to Amazon S3 bucket because the bucket was misidentified or that permissions were configured incorrectly.

Navigator Audit Server

"No serializer found" error

**Symptom:** Selecting an Audits page results in error message.

**Error message:**

```
No serializer found
```

**Possible cause:** Communication failure somewhere between Navigator Metadata Server and Cloudera Manager Server and its communication to Navigator Audit Server. This is an existing Known Issue that needs to be resolved so that the error message can be properly mapped.

**Workaround:** This workaround does not resolve the issue, but provides additional information as to the actual source of the underlying error.

1. Log in to the Cloudera Navigator console.
2. Enable Navigator API logging.
3. Perform your action on the Audits page that raised the error message and see if the underlying issue is caught in the API log. If the Navigator API logging reveals no additional information, turn on API logging for Cloudera Manager:
   - Log in to the Cloudera Manager Admin Console.
   - Select Administrator > Settings
• Select **Advanced** for the **Category** filter.
• Click the **Enable Debugging of API** check box to select it. The server log will contain all API calls. Try your request again and see if the source error message displays in the response.

**Processing a backlog of audit logs**

**Problem:** You have logs that include audit events that were not processed by the Cloudera Navigator Audit Server, such as logs for events that took place before audit server was online or during a period when audit server was offline.

**Solution:** A backlog of logs can be processed by audit server as follows:

1. Determine what days were not processed and will need to have partitions created for them. The audit log files have the UNIX epoch appended to their name:

   ```
   hdfs-NAMENODE-341ad9b94435839ce45b8b22e7c805b3-HdfsAuditLogger-1499844328581
   ```

   You want the dates from the oldest and newest files. You can do this by sorting the list and identifying the first and last files. For example, for HDFS audit files:

   - **The oldest:** $ ls ./hdfs* | head -1
   - **The newest:** $ ls ./hdfs* | tail -1

   Depending on your shell, you can convert the epoch value to a date using `date -r epoch`.

2. Create partitions in the Navigator Audit Server database for days that have missing audits.

   Create a partition from an existing template table using SQL commands. Use `SHOW TABLES` to list the template tables. For example, the template table for HDFS is `HDFS_AUDIT_EVENTS`.

   Using MySQL as an example, if you have partitions up to March 31, 2017 and your audit logs include HDFS data for the first week of April, you would run the following SQL commands:

   ```
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_01 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_02 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_03 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_04 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_05 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_06 LIKE HDFS_AUDIT_EVENTS;
   CREATE TABLE HDFS_AUDIT_EVENTS__2017_04_07 LIKE HDFS_AUDIT_EVENTS;
   ```

3. Change the Navigator Audit expiration period to be longer than the time period of the logs you want to process.

   Set the expiration period in Cloudera Manager. Search for the property “Navigator Audit Server Data Expiration Period”.

4. Stop the Cloudera Manager agent on the host running the role for the service whose logs will be restored.

   For example, if the logs are for HDFS, stop the agent on the namenode. For HiveServer2, stop the Cloudera Manager agent on the node where HiveServer2 is running. See [Starting, Stopping, and Restarting Cloudera Manager Agents](#).

   If you aren’t sure of which host is involved, the log files contain the UUID of the role which generated them. Go to the host defined for that role. For high-availability clusters, make sure that you stop the role on the active host.

5. On that host, copy the backed up audit logs to the audit directory.

   The location is typically `/var/log/service/audit`. For example, `/var/log/hadoop-hdfs/audit`. The specific locations are configured in the Cloudera Manager `audit_event_log_dir` properties for each service.

6. Move the WAL file out of this directory. (Back it up in a safe location.)

7. Restart the Cloudera Manager agent.

   On restart, Cloudera Manager agent will not find the WAL file and will create a new one. It will process the older audit logs one by one.
8. Verify that the audits appear in Navigator.
   You can also check the contents of the WAL file to see which logs were processed.

Navigator Metadata Server

"Missing required field: id" (Solr exception)

Symptom: After removing the Navigator Metadata Server storage directory (the `datadir`), an error message for the `id` field displays.

Error message:
```
[qtp1676605578-63]: org.apache.solr.common.SolrException: [doc=0002ac6b75cb17721ad8324700f473cb] missing required field: id
```

Possible cause: This error indicates that the Navigator Metadata Server database and storage directory are out-of-sync because of a change that was made in the Cloudera Navigator 2.9 (Cloudera Manager 5.10) release. When the Navigator Metadata Server role starts, it compares state information in the database with the same information in the storage directory. This error can appear if the storage directory has been deleted but the database has not.

Steps to resolve: If this error message occurs and the storage directory was recently deleted, you can remove the stale state information from the database. To do this:

1. Take note of the Navigator Metadata Server Storage Dir directory.
   Log into Cloudera Manager and browse to Cloudera Management Services > Configuration > Navigator Metadata Server > Navigator Metadata Server Storage Dir. Note the storage directory location.

2. Stop the Navigator Metadata Server role by navigating to Cloudera Management Service Instances > Navigator Metadata Server > Actions > Stop.

3. Log in to the database instance configured for use with Navigator Metadata Server.

4. Run the following SQL commands:
```sql
delete from NAV_UPGRADE_ORDINAL;
insert into nav_upgrade_ordinal values (-1, -1);
```


The Navigator Metadata Server role should restart successfully.

Repairing metadata in the storage directory after upgrading

If you ran Cloudera Navigator on Cloudera Manager versions 5.10.0, 5.10.1, or 5.11.0, you should upgrade to a later release immediately to avoid a known software problem as reported in the knowledge base entry "Upgrade needed for Cloudera Navigator included with Cloudera Manager releases 5.10.0, 5.10.1, and 5.11.0 due to high risk of failure through exhausting data directory resources." This document describes some additional steps to perform after upgrade to mitigate for duplicate and potentially broken relation metadata added to the Navigator storage directory because of the broken versions.

To fully recover the Navigator storage directory:

1. Upgrade Cloudera Manager to 5.10.2, 5.11.1, or later releases.
   See Upgrading Cloudera Manager.

2. Set properties to enable background repair tasks.
   From Cloudera Manager, go to Cloudera Management Service > Navigator Metadata Server > Advanced.
Enter the following configurations to the "Navigator Metadata Server Advanced Configuration Snippet (Safety Valve) for cloudera-navigator.properties" property.

```javascript
nav.backend.task.TRIGGER_HIVE_TABLES_EXTRACTION.enabled=true
nav.backend.task.HDFS_PC_RELATIONS_BUILDER.enabled=true
nav.backend.task.HDFS_MR_RELATIONS_BUILDER.enabled=true
nav.backend.task.PIG_HDFS_RELATIONS_BUILDER.enabled=true
```

3. Restart the Navigator Metadata Server role.

The full cycle of cleaning up the relations may take days or weeks depending on the number of relations affected.

These instructions correspond to “Repair metadata” in the following decision tree:
Appendix: Apache License, Version 2.0

SPDX short identifier: Apache-2.0

Apache License
Version 2.0, January 2004
http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License.

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.


Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims
licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution.

You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

1. You must give any other recipients of the Work or Derivative Works a copy of this License; and
2. You must cause any modified files to carry prominent notices stating that You changed the files; and
3. You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
4. If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions.

Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement You may have executed with Licensor regarding such Contributions.

6. Trademarks.

This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty.

Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability.

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability.
END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets "[]" replaced with your own identifying information. (Don't include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same "printed page" as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.