

Cloudera Runtime 7.0.3

## Atlas Use Cases

Date published: 2019-11-22

Date modified:

# CLOUdera

<https://docs.cloudera.com/>

# Legal Notice

© Cloudera Inc. 2025. All rights reserved.

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

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

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

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

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

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

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

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

# Contents

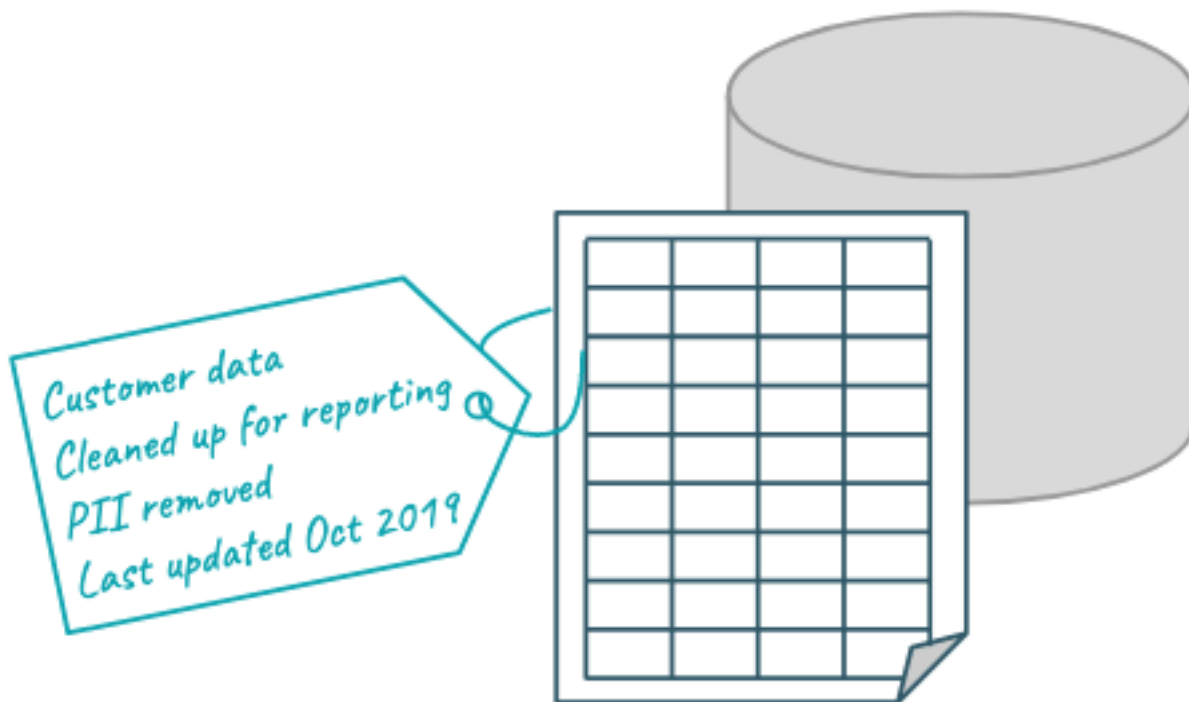
|                                                |          |
|------------------------------------------------|----------|
| <b>Governance overview.....</b>                | <b>4</b> |
| Data Stewardship with Apache Atlas.....        | 5        |
| Atlas dashboard tour.....                      | 6        |
| Apache Atlas metadata collection overview..... | 8        |
| Atlas Metadata model overview.....             | 10       |

## Governance overview

Concepts for collecting, creating, and using metadata.

### What is Apache Atlas?

Atlas is a metadata management and governance system designed to help you find, organize, and manage data assets. Atlas creates “entities” or metadata representations of objects and operations in your data lake. You can add business metadata to these entities so you can use business vocabulary to make it easier to search for specific assets.



### Apache Atlas uses metadata to create lineage relationships

Atlas reads the content of the metadata it collects to build relationships among data assets. When Atlas receives query information, it notes the input and output of the query and generates a lineage map that traces how data is used and transformed over time. This visualization of data transformations allows governance teams to quickly identify the source of data and to understand the impact of data and schema changes.

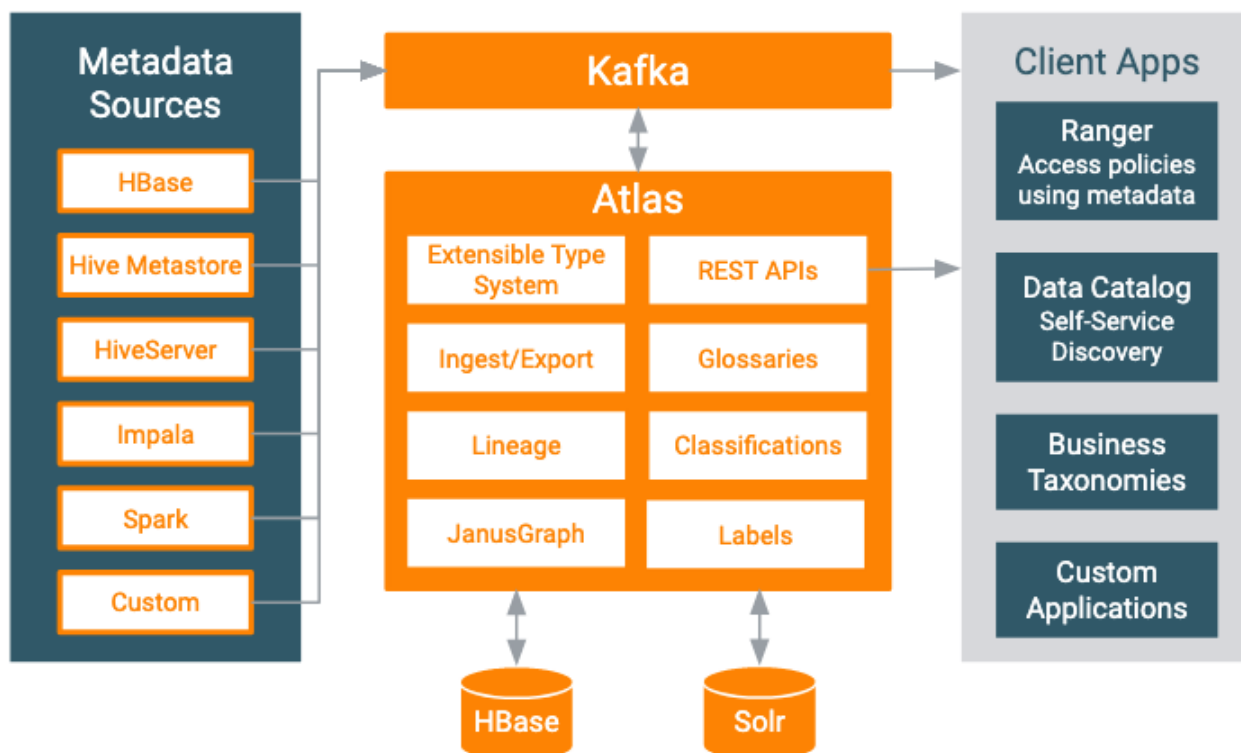
### Adding to entity metadata makes searching easier

Atlas manages *classifications* and *labels* that you create and use to enhance the metadata for data assets. You can create and organize classifications and labels to use for anything from identifying data cleansing stages to recording user comments and insights on specific data assets. When you use classifications, the Atlas Dashboard makes it easy to search, group, report, and further annotate the entities you label. Classifications themselves can be organized into hierarchies to make them easier to manage.

Atlas also provides an infrastructure to create and maintain business ontologies to label your data assets. Atlas’ “glossaries” include “terms” so you can build agreed-upon lists for department- or organization-wide vocabulary to identify and manage data. Adding a term gives you a single-click report of entities identified by that term.

## Apache Atlas architecture

Atlas runs as an independent service in a Hadoop environment. Many Hadoop data processing and storage services include Atlas add-ons that publish metadata for the services' activities to a Kafka message topic. Atlas reads the messages and stores them in JanusGraph to model the relationships among entities. The datastore behind JanusGraph is HBase. Atlas stores a search index in Solr to take advantage of Solr's search functionality.



Pre-defined hooks exist for Hive, Impala, Kafka, NiFi, Spark, and Sqoop.

Atlas also provides “bridges” that import metadata for all of the existing data assets in a given source. For example, if you start Atlas after you’ve already created databases and tables in Hive, you can import metadata for the existing data assets using the Hive bridge. Bridges use the Atlas API to import the metadata rather than publishing messages to Kafka.

If you need a hook or bridge to automate collecting metadata from another source, use the Atlas Java API to create a custom Atlas addon.

## Data Stewardship with Apache Atlas

Concepts for collecting, creating, and using metadata.

The value of Atlas metadata for organizing and finding data increases when you augment the generated “technical” metadata by using your organization's business vocabulary. Here’s how you would do that:

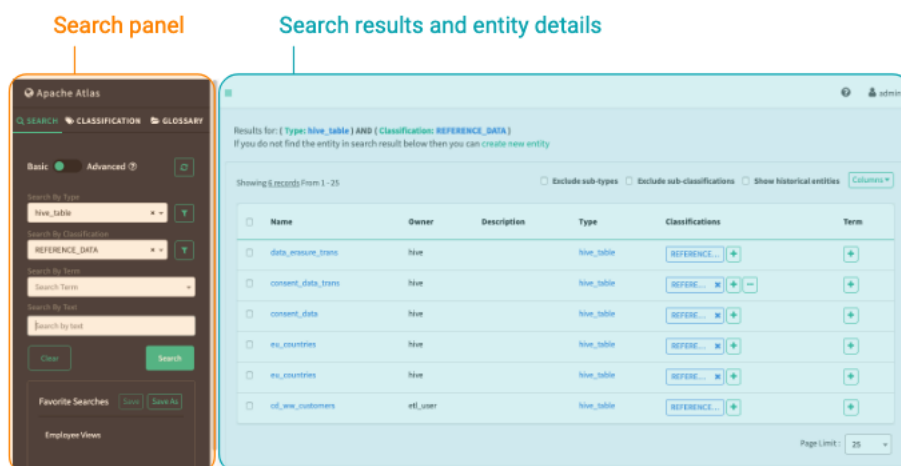
- **Tools.** You can add metadata to entities using labels, classifications, attributes, and a glossary of terms. The glossary allows you to identify synonyms so that the vocabulary from different teams doesn't get in the way of identifying the same data.
- **Planning.** Figure out who and how to apply the tools: set up an overall plan for what kinds of metadata you want to apply, design some conventions for how to apply them and who can apply them. Design some processes to oversee metadata as it collects to make sure the results are consistent and useful; identify synonyms and antonyms.
- **Examples.** This document includes examples of ways you can organize your metadata; strategies that describe how to optimize for specific use cases.

## Atlas dashboard tour

Quick introduction to the Atlas user interface and terms.

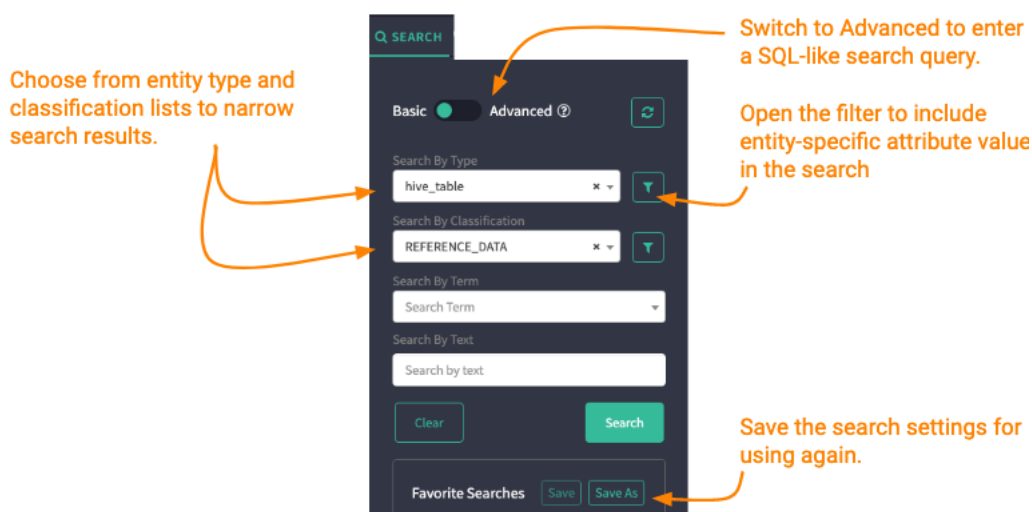
The Atlas user interface is a metadata dashboard with two parts:

- Search panel on the left.
- Detail panel on the right where search results appear, and, when you drill into a specific entity, the details for that entity are displayed. Each detail page has a header section and a series of tabbed panels, all of which are oriented to the metadata for that entity type.



## Searching

The search panel has three tabs for searching: the general Search tab, and predefined searches based on Classifications and Glossary terms. In the general Search tab, you choose from existing lists of metadata types to narrow the search results. Switching to the Advanced search lets you enter specific search queries; both basic and advanced searches can be saved for easy reuse.



In the Classification tab, selecting a classification displays all the entities that are marked with that classification. Find a specific classification using the search box or browse through the classification hierarchy that you define when you create classifications.

In the Glossary tab, selecting a term displays all the entities that are marked with that term. Find a specific term using the search box or browse through terms by glossary. You can also find specific terms using the category view:

browse through the hierarchy you build of your organization's business glossary; when you select a category, the display pane shows the terms assigned to the category. When you select one of those terms, Atlas displays the entities associated with that term.

When you run a search and Atlas returns results, you see a paged-list of entities that match the search criteria. From here, you can go back to the search options and further refine your search or use controls to change how the search results are presented.

The screenshot shows the Atlas search results interface. At the top, a search bar contains the criteria: **( Type: hive\_table ) AND ( Classification: REFERENCE\_DATA )**. Below the search bar, a message states: "If you do not find the entity in search result below then you can [create new entity](#)".

Below the message, there are filters and controls:
 

- Showing 6 records From 1 - 25**
- Checkboxes for **Exclude sub-types**, **Exclude sub-classifications**, and **Show historical entities**.
- A **Columns** dropdown menu.

The main table displays search results with the following columns: **Name**, **Owner**, **Type**, **Classifications**, and **Term**. The table contains six rows of results, each with a checkbox in the **Name** column and a **+** icon in the **Term** column.

Annotations with arrows point to specific features:
 

- Search criteria**: Points to the search bar at the top.
- Remove some entity types from the search results by excluding:**
  - sub-entities,
  - sub-classifications, or
  - entities marked as deleted
 Points to the filter checkboxes.
- Open Columns to change what attributes are shown in the search results**: Points to the **Columns** dropdown.
- Select entities to apply the same action to multiple entities**: Points to the checkboxes in the **Name** column.
- To associate classifications or terms to entities, click **+****: Points to the **+** icons in the **Term** column.

## Viewing entity details

When you click a link for an entity in the search results, Atlas opens an entity detail page that includes the metadata collected for the entity. The detail page organizes the entity content in tabs:

### Properties

The Properties tab includes the system metadata collected for this entity and any user-defined properties added. It also contains a list of labels applied to the entity. You can use values from any of the "string" data type properties to find this entity using free-text search.

### Lineage

Atlas UI displays a lineage graph for each entity. The graph appears in the Lineage tab in the entity details page. For data asset entities, the lineage graph shows when the entity was an input or an output from an operation. For process entities, the lineage graph shows all input and output entities used by or produced by the operation.

### Relationships

This tab lists the other entities that are associated with this entity as "relationships". You can show the list of related entities as a list or as a graph. Use this tab to navigate among entities. The special relationship types "input" and "output" include the entities that make up lineage.

### Classifications

The Classifications tab shows the classifications associated with this entity (which are also shown in the top section of the detail page). It allows you to add, update, or remove classifications from the entity. Note that to manage classifications, you need to be granted privileges to perform classification actions.

### Audits

Atlas records the changes that occur to entity metadata. The changes are listed in the Audit tab in the entity details page. Audits show when Atlas updates the entity's metadata, including the following changes:

- Classifications added or removed
- Entity attributes updated
- Labels added, updated, or removed
- Relationships added, updated, or removed
- Glossary terms added or removed

### Schema

When the current entity is a table, the Schema tab appears and lists columns in the table. Use this tab to drill into a specific column or to add classifications to columns (no need to open the detail page for the column to add a classification).

## Apache Atlas metadata collection overview

Actions performed in cluster services create metadata in Atlas.

Atlas provides addons to many Hadoop cluster services to collect metadata when the service performs certain operations. The Atlas addon or "hook" assembles a predefined set of information and sends it to the Atlas server. The Atlas server reads through the metadata and creates entities to represent the data sets and processes described by the metadata. Atlas may create one or many entities for each event it processes. For example, when a user creates a namespace in HBase, Atlas creates a single entity to represent the new HBase namespace. When a user runs a query in HiveServer, Atlas may create many entities, including entities to describe the query itself, any tables involved in the query, entities for each column for each table involved in the query, and so on.

The following table lists the services that are integrated with Atlas by default. For each service, the table lists the events produced by the service that Atlas acknowledges and the entities Atlas produces in response to each event. Note that there isn't always a one-to-one relationship between the event and an entity: the entities produced from a single event depend on the event itself.

| Source     | Actions Acknowledged                                                   | Entities Created/Updated                                                                                                        |
|------------|------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| HiveServer | ALTER DATABASE<br>CREATE DATABASE<br><br>DROP DATABASE                 | hive_db, hive_db_ddl                                                                                                            |
|            | ALTER TABLE<br>CREATE TABLE<br>CREATE TABLE AS<br>SELECT<br>DROP TABLE | hive_process, hive_process_execution, hive_table, hive_table_ddl, hive_column, hive_column_lineage, hive_storagedesc, hdfs_path |

| Source  | Actions Acknowledged                                                                                          | Entities Created/Updated                                                                           |
|---------|---------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
|         | ALTER VIEW<br>ALTERVIEW_AS_SELECT<br>CREATE VIEW<br>CREATE VIEW AS<br>SELECT<br>DROP VIEW                     | hive_process, hive_process_execution, hive_table, hive_column, hive_column_lineage, hive_table_ddl |
|         | INSERT INTO (SELECT)<br>INSERT OVERWRITE                                                                      | hive_process, hive_process_execution                                                               |
| HBase   | alter_async                                                                                                   | hbase_namespace, hbase_table, hbase_column_family                                                  |
|         | create_namespace alter_namespace<br>drop_namespace                                                            | hbase_namespace                                                                                    |
|         | create table<br>alter table<br>drop table<br>drop_all tables                                                  | hbase_table, hbase_column_family                                                                   |
|         | alter table (create column family)<br>alter table (alter column family)<br>alter table (delete column family) | hbase_table, hbase_column_family                                                                   |
| Impala* | CREATETABLE_AS_SELECT                                                                                         | impala_process, impala_process_execution, impala_column_lineage, hive_db<br>hive_table_ddl         |
|         | CREATEVIEW                                                                                                    | impala_process, impala_process_execution, impala_column_lineage, hive_table_ddl                    |
|         | ALTERVIEW_AS_SELECT                                                                                           | impala_process, impala_process_execution, impala_column_lineage, hive_table_ddl                    |
|         | INSERT INTO<br>INSERT<br>OVERWRITE                                                                            | impala_process,<br>impala_process_execution                                                        |
| Spark*  | CREATE TABLE USING<br>CREATE TABLE AS<br>SELECT,<br>CREATE TABLE<br>USING ... AS SELECT                       | spark_process                                                                                      |
|         | CREATE VIEW AS SELECT,                                                                                        | spark_process                                                                                      |
|         | INSERT INTO (SELECT),<br>LOAD DATA [LOCAL]<br>INPATH                                                          | spark_process                                                                                      |

\*For these sources, Atlas collects the corresponding asset metadata from HMS. Atlas reconciles the entity metadata received from Kafka messages from each source.

#### Related Information

[Hive Server 2 metadata collection](#)

[HBase metadata collection](#)

[Impala metadata collection](#)

## Atlas Metadata model overview

Atlas' model represents cluster data assets and operations, and is flexible enough to let you represent objects from other sources.

The flexibility Atlas' metadata model lets you represent the objects and relationships among them so you can produce a map of your data lake. Atlas lets you create new instances of predefined entity types and lets you define new types of entities so you can represent data assets and actions from additional data sources or even services that do not reside in Hadoop. Atlas' building blocks are entities, relationships, classifications, enumerations, and structures.

Entities are a collection of attributes that model or represent a data asset or data action. Entities are the unit that Atlas returns in search results or shows as nodes in a lineage diagram. You use the Atlas API to change the model to add attributes to entity definitions. Labels are modeled as attributes on a given entity instance; you can add user-defined properties to individual entity instances (without affecting the entity type definition).

Relationships describe connections between two entities. You can create relationship definitions with custom attributes to represent behaviors that are specific to your processes. Changes to relationship definitions require changing the model through the Atlas API.

Classifications are named sets of key-value pairs that can be associated with entities. Classifications are distinct from entity attributes:

- Classifications are not part of entity metadata so they are a way to add metadata to entities without updating entity type definitions.
- Classifications can be added to any entity type.
- Atlas can propagate classifications through lineage relationships.
- Classifications can be used in Ranger to drive access policies.

Atlas supports defining custom enumerations and data structures as well, similar to those constructs in structured programming languages. Enums can be used in attribute definitions to store lists of predetermined values; structs can be used in attribute definitions to identify more complex data types.