Cloudera Runtime 7.2.15

Atlas Search

Date published: 2019-09-23
Date modified: 2022-05-12

https://docs.cloudera.com/

Legal Notice

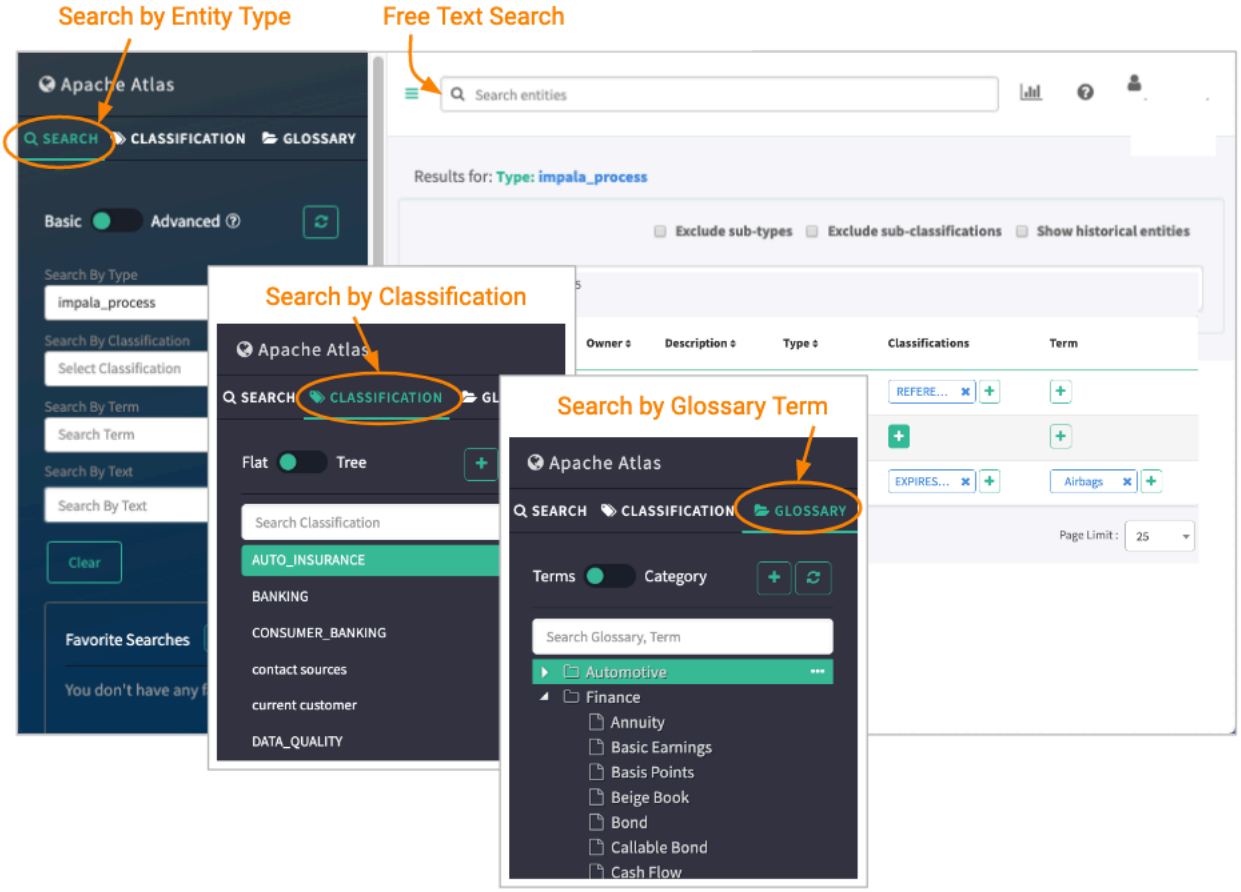
Contents

Searching overview

Search using any metadata text, by entity type, entity and system attribute value, classification, or glossary term.

You can search for entities using four search modes:

- Free text from all string-type metadata, including classifications, labels, Business Metadata attributes, and user-defined properties
- By entity type, with refinements by system attribute, entity attribute, Business Metadata attribute, classification, term, label, or user-defined property
- By classification
- By glossary term



Related Information

Using Free-text Search
Using Basic Search
Using Search filters
Searching for entities using classifications
Searching for entities using Business Metadata attributes
Searching for entities using terms
Using advanced search
Saving searches

Using Basic Search

Search using an entity type in Basic Search.

With Basic Search, Atlas returns all of the entities of the type you select.

There are many ways you can define a Basic Search. Setting a value in more than one search field builds a logical AND condition for the search. To repeat the same search, click the Refresh button. 

Search By Type

- Choose an entity type to limit the search.
- Choose _ALL_ENTITY_TYPES to apply an attribute filter across all entity types.

Search By Type and specify attribute values using the Filter 

The Attribute Filter dialog box lists all the attributes that correspond to the selected entity type, including:

- Technical attributes specific to the entity type
- System attributes, including classifications, labels, and user-defined properties
- Business Metadata attributes
- Terms

Note: When searching on classifications in the Search By Type filter, use "contains" rather than "=" for the filter operator. If there is more than one classification assigned to an entity, "contains" matches a single classification name; equal only matches the entire list of classifications names.

Search By Classification

- Choose an existing classification; the search returns all entities that have that classification assigned to them.
- Choose _ALL_CLASSIFICATION_TYPES to apply an attribute filter across all classifications.
- Choose _CLASSIFIED or _NOT_CLASSIFIED with an entity type selected to find entities of that type with any or no classifications assigned.

Search By Classification and specify attribute values using the Filter 

The Attribute Filter dialog box lists all the attributes for the selected classification; set a value to one or more attributes to define the search. You can choose to match partial strings using the "contains", "begins with", and "ends with" operators.

Search by Term

Choose an existing glossary term. You can enter the first few letters to select a term from a list of matching terms. This filter is case-sensitive.

Search by Text

Search on string values for technical, system, Business Metadata, and classification attribute values. Labels and terms are also included. This search is the same as the Free-Text search; note that when you enter text in the Free-Text search box, it fills in this Search By Text field also.

You can also save these searches when they are useful to run more than once.

Related Information

Using Free-text Search
Searching for entities using Business Metadata attributes
Searching for entities using terms
Searching for entities using classifications
Saving searches
Apache Atlas metadata attributes
Using Search filters

Using Search filters

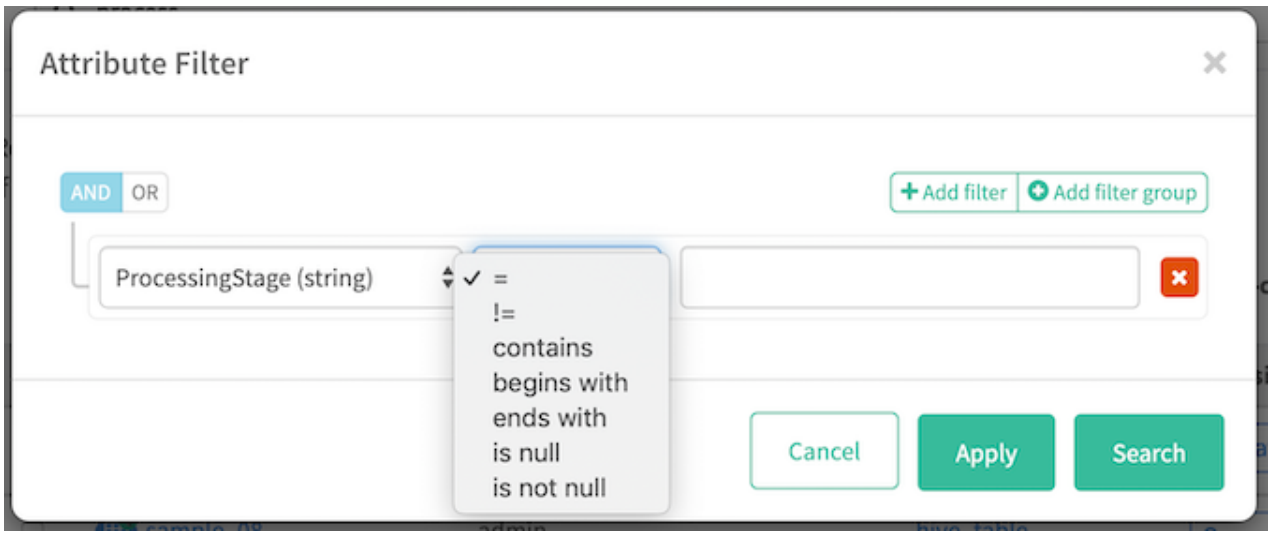
The Basic Search panel includes filter icons that allow you to search for entities based on one or more attribute values.

In a filter row, the attribute data type determines which of the following operators can be used to define your search criteria:

Strings	Dates	Enumerations Boolean	Numerics
=		=	=
!=		!=	!=
	>	>	>
	<	<	<
is null	is null	is null	is null
is not null	is not null	is not null	is not null
contains			
begins with			
ends with			

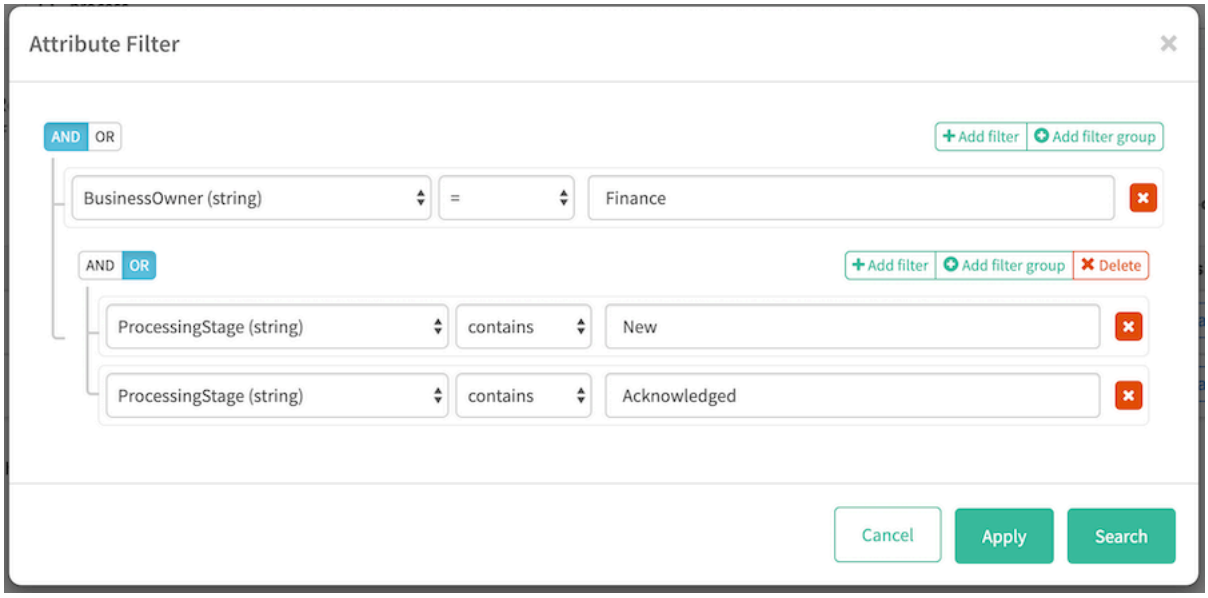
All classification attributes are string values; numerics include byte, short, int, float, double, and long attribute data types.

Note: If the attribute you are searching for could include multiple values, use "contains" rather than "=" to make sure the search finds the individual value out of the list.

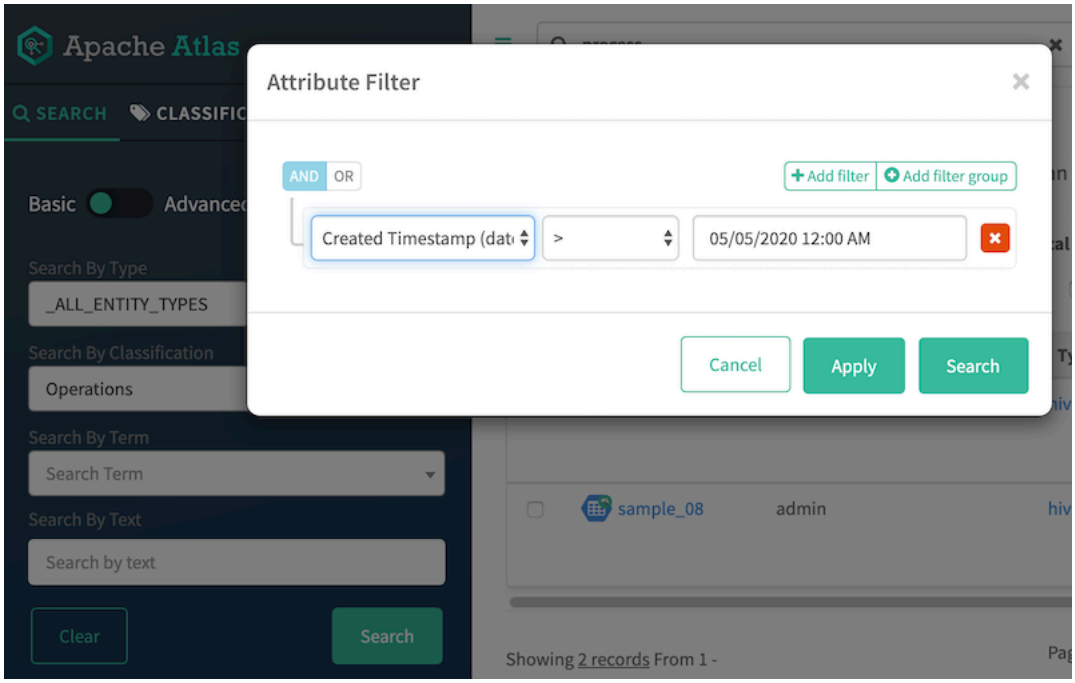


To search on values for more than one attribute, add another filter row to the search filter (click Add filter). The search can find entities matching either filter criteria (logical OR) or matching both criteria (logical AND). Set the logic using the AND / OR buttons at the top-left of the filter rows.

You can combine logical AND and OR criteria using filter groups. The logic is the same within a filter group; use more than one filter group to produce both AND and OR logic. For example, the following Classification attribute filter searches for entities that are at "new" or "acknowledged" stages in their processing and are owned by the Finance business team.



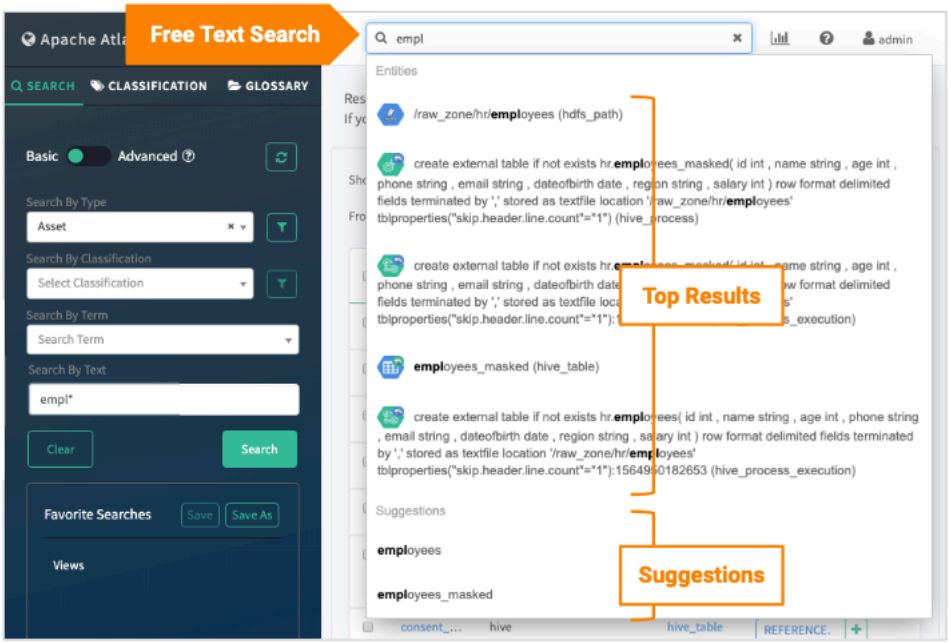
If you wanted to further limit the search results to the entities that were created in Atlas in the last 24 hours, you would open the attribute filter for Search by Entity Type and set the system attribute "Created Timestamp" less than 24 hours. To open the Search by Entity Type filter, you would need to select an entity type or "_ALL_ENTITY_TYPES".



Using Free-text Search

Apache Atlas builds a ranked index of metadata values so you can search for values across all metadata.

The search field in the top of the dashboard lets you search for entities, classifications, or terms by entering any full or partial text to match any entity metadata values. Atlas searches all metadata fields that have string data types, so you can use this search tool to find entities by their labels, descriptions, locations, or other metadata.



Searches are case insensitive. Searches automatically append the asterisk (*) wildcard to the search term or terms so that the search behavior is to find partial strings anywhere they occur in the metadata value. The following single and double characters have special meaning:

```
+ - && || ! ( ) { } [ ] ^ " ~ * ? : \
```

If your search string includes one of these characters, surround the string in double quotation marks or escape the special character with a backslash.

You can see that the search terms you use in the search at the top of the dashboard are also inserted into the free text search field in the left Search panel: you can combine the free text search with other selections to narrow the search results. The combination acts as an “AND” with other search criteria.

Search result ordering: The search results are ranked by how well they match the search terms, with entities that match on more than one metadata value being ranked higher.

Different metadata have different scores, where the highest scoring metadata fields are entity names, including Kafka topic names. Descriptions, users/owners, query text, and comments rank next. Locations, namespaces, domains, etc. come next. Search results are not ordered in any specific way among results that have the same search ranking.

Suggestions: As you enter your search text, you see the five highest-ranked matching items and as many as five suggestions.

The matching items are ranked in the same way as the general search results, case-sensitive (at the moment) terms that “contain” the search terms; If there are more than five search results with the same search ranking, the five shown are randomly ordered from the highest scoring results.

The suggested items are chosen from search results that match with a “starts with” behavior.

Ignore or Prune pattern to filter Hive metadata entities

Atlas supports metadata and lineage updates from services like HBase, Hive, Impala, and Spark.

These updates are in the form of messages that are posted by these services. The messages contain Atlas entities specific to the service. The notification processing module within Atlas processes these messages.

Typically, most of the metadata is tracked. Sometimes, a part of the schema changes more often than not and tracking these frequent changes creates metadata that is insignificant. The Atlas notification processing system gets overloaded with the frequently changing schema updates. The resultant outcome might be that the low-value messages are processed at the expense of messages that contain critical schema updates.

To overcome such a pattern within a data processing pipeline, you can employ a couple of options:

- Ignore schema updates.
- Preserve an abbreviated form of the entity.

The Ignore and Prune feature within Atlas addresses this scenario for Hive Metastore and Hive Server2 (HS2) hooks. This feature is a mechanism to specify which Hive tables should be ignored and which ones should be pruned. This feature helps regulate data that is posted to Atlas. The user is able to choose data that is important for metadata management and lineage capture.

Note: This mechanism does not exist for other hooks.

Attention: The Ignore / Prune configurations feature is not supported when the configurations are provided in upper case or mixed case. You must use the lower case while setting up the Ignore / Prune configurations.

Tables whose lifecycle is of no consequence are targeted for being ignored. Tables whose lifecycle need not be tracked closely or for garnering minute details are targeted for pruning.

Attention: Atlas tracks the table and table-level lineage; however, columns of pruned table and their column level lineage are not tracked in Atlas.

Use case

As a part of the Extract/Transform/Load (ETL) data pipeline, services such as Hive use a number of temporary and/or staging tables that are short-lived. These temporary and/or staging tables are generally employed during the extract or transform phase before the data is loaded. Once the processing is complete, these tables are not used anymore and are deleted.

With Atlas Hive Hook enabled, Atlas captures metadata events, lifecycle, and lineage of all the Hive entities.

Temporary tables that are created only to aid the development process are safe to be ignored. Metadata for these tables are not generated or reported into Atlas.

For staging tables, tracking details like columns and column-lineage in Atlas may not be useful. By not tracking the information in Atlas, it can significantly reduce the time it takes to process notification and can help the overall performance of Atlas.

You can ignore temporary tables completely. Just the minimum details of these staging tables can be stored in Atlas, to capture data lineage from source to target table through all the intermediate staging tables.

Setting Ignore and Prune Properties

The ignore and prune configuration properties can be set both at Atlas server-side and Hive hooks configuration.

Setting it at Hive Hook side prevents Atlas' metadata from being generated.

If the metadata for ignored and pruned elements is generated and posted on Atlas' Kafka topic, setting this property on Atlas' server side handles these elements before they get stored within Atlas.

Both these properties accept Java regex expressions. For more information, see documentation.

How Ignore and Prune feature works

The configurations are matched against the Hive table's qualifiedName attribute.

Within the Hive hook, qualifiedName attribute value has this format: database.table@namespace

The namespace is the value specified by the atlas.metadata.namespace property.

For example, for a Hive hook, the property atlas.metadata.namespace is set to glv.

On that server, for a table t1 which is a part of database db1, the qualifiedName value is: db1.tb1@glv

Ignore Pattern

Hook-side

atlas.hook.hive.hive_table.ignore.pattern

Atlas server side

atlas.notification.consumer.preprocess.hive_table.ignore.pattern

Prune pattern

Hook-side

atlas.hook.hive.hive_table.prune.pattern

Atlas server side

atlas.notification.consumer.preprocess.hive_table.prune.pattern

Using Ignore and Prune patterns

You can configure both Ignore and Prune patterns to manage your data.

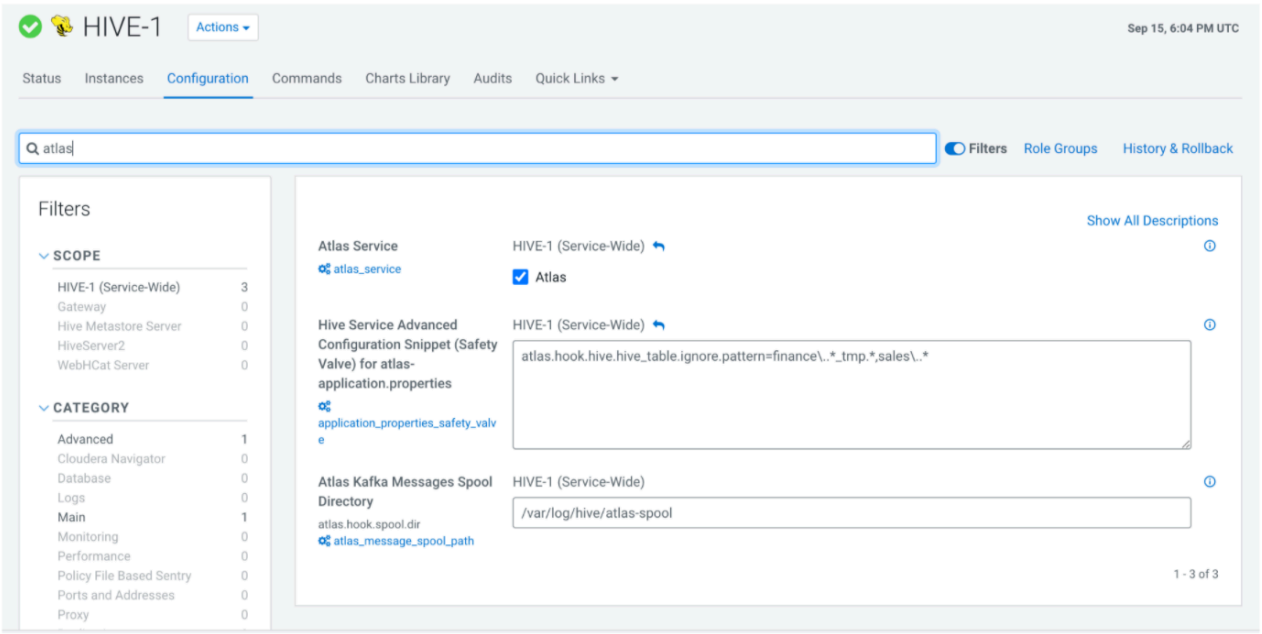
Using the Ignore pattern

Atlas ignores temporary managed tables by default. But an external temporary table is captured because the table uses the HDFS path for storage and Atlas creates a lineage in between.

To disregard the temporary table and avoid Atlas processing it, you can set up appropriate configurations in Hive and Atlas and later restart the services.

For example, if all tables in the ‘sales’ database and the tables that contain ‘_tmp’ in the ‘finance’ database should be ignored, the property can be set as follows in your Cloudera Manager instance.

Hive Metastore Server and Hive settings:



atlas.hook.hive.hive_table.ignore.pattern=finance\..*_tmp.*,sales\..* is set in Cloudera Manager Hive Service Advanced Configuration Snippet (Safety Valve) for atlas-application properties in Hive(HMS) and Hiveserver2.

Atlas server

atlas.notification.consumer.preprocess.hive_table.ignore.pattern=finance\..*_tmp.*

With the above configurations, tables having _tmp in their names, in the finance database are ignored.

Note: The "." is a special regular expression character, hence had to be escaped with a backslash (\).

Using the Prune pattern

Staging tables are created to hold data temporarily during a query execution and are manually dropped once the processing is completed. It might be insignificant to track the details of the staging tables.

For example, in the below images, the finance table contains 333 columns (column names blurred) and the staging tables are created frequently by running an “INSERT OVERWRITE TABLE” query on the finance table. Processing is executed on the data in staging tables and later the staging tables are deleted as observed in the table level lineage diagram.

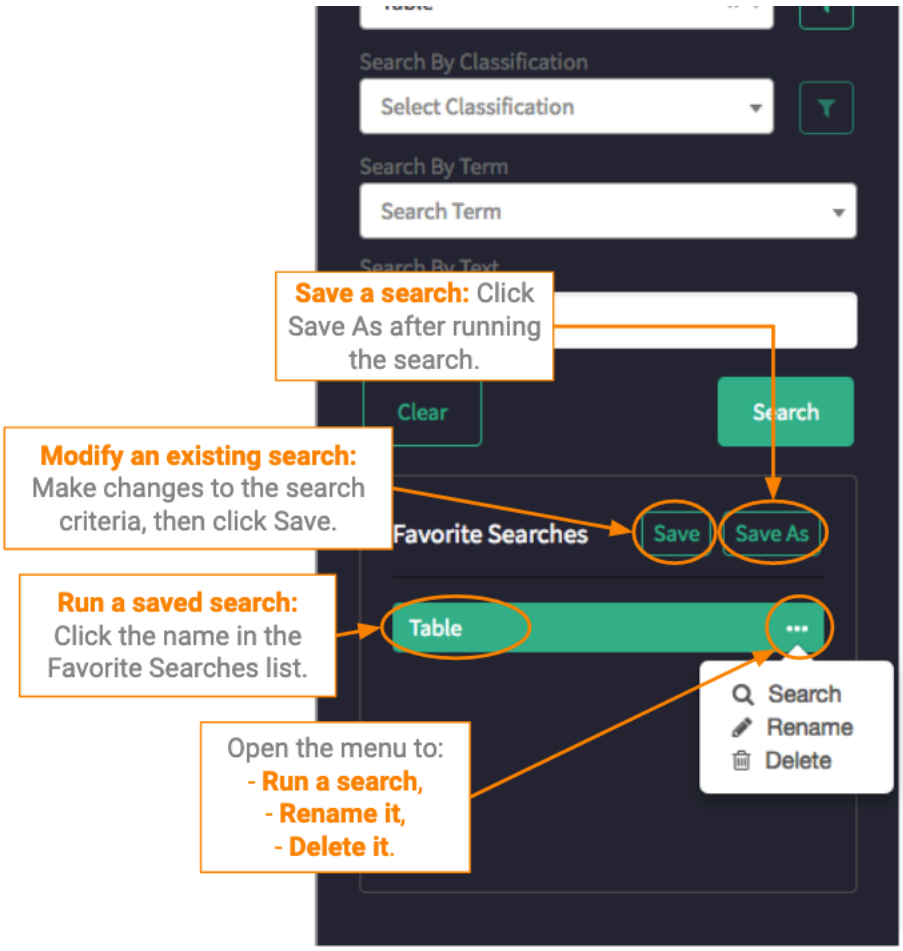
Every time you run a query to create lineage between tables, column-level lineage is also created along with table-level lineage.

Note: If the query involves all the columns, 333 hive_column_lineage entities are created and pushed to the ATLAS_HOOK Kafka topic.

Saving searches

Saving a search saves the search criteria with a name that will help you remember what the search returns.

After you run a search, you can save it under a name in the list of Favorites. Here's what you can do to save a search and to use a search you've already saved:

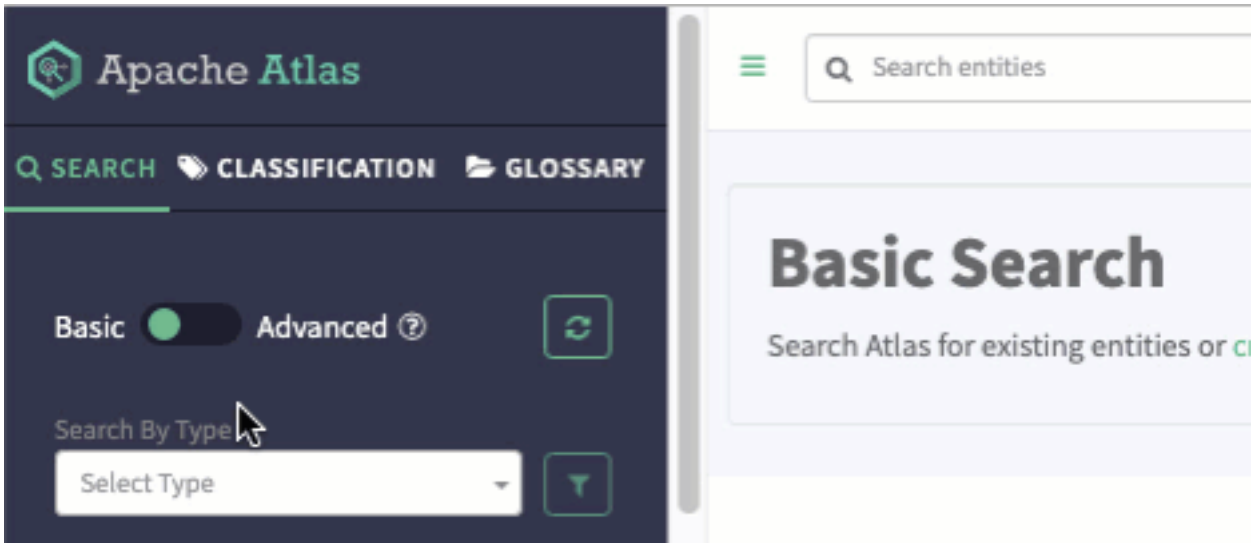


Using advanced search

Apache Atlas advanced search lets you use a query language to combine criteria and refine search results.

Advanced search gives you more control over search criteria through the Atlas domain-specific query language.

In the left navigator pane, Search tab, switch to Advanced search mode by sliding the green toggle button from Basic to Advanced.



Select an entity type if appropriate, then add your query to refine the search results. Here are some examples of advanced search queries:

- Search for partial names

```
from hive_table where name LIKE '*_dim'
```

- Search in date ranges

  Note that the entity attributes may contain date fields that are populated from the source while the system attributes contain date fields that are populated when the entity is created in Atlas. The values can be different.

Entity attribute createTime	`from hive_table where createTime > '2019-01-01' and createTime < '2019-01-03'`
System attribute Create Timestamp	`from hive_table where __timestamp > '2019-01-01' and __timestamp < '2019-01-03'`

- Search for deleted entities

  System attributes (with two underscores before the name) are available on all entity types.

```
from hive_table where __state = DELETED
```

- Search for multiple criteria

```
from hdfs_path where isFile = true and name = "Invoice"
```

- Return specific metadata

```
from hive_table where name = 'customer_dim' select owner, name,
        qualifiedName, createTime
```

- Search for entities with classifications

```
from hive_table where hive_table isa Dimension select owner, name, quali
fiedName
```

See the advanced search reference for information about the query language and for more examples.

Improved search capabilities for Glossary terms and Relationships

In Atlas, while using the Advanced Search feature, you can now search for entities based on the glossary term, by using the newly introduced hasTerm keyword that allows users to search the entities which are tagged with them. You can also search the entities based on relationship attributes using the where clause.

In order to search for those entities having a specific glossary term, you must add a fully qualified name. For example: {termName}@{glossaryName}. This term gets compared with the qualifiedName attribute of glossary type.

Where as, when you add only the term name, the resultant output will be the available entities with the specific term name. This is irrespective of what type of glossary it is in and would compare with the name attribute of the glossary type.

Additionally, to search for entities related to the referenced entities, you must add the relationship attribute and value to search for in thewhere clause. For example: To search for tables under a specific database. For example: {relatio nshipName}.{attributeName} = {value}

Examples of Glossary term filtering:

- Table hasTerm savingAccount1234
- Table hasTerm “savingAccount1234@Banking"
- Table hasTerm “savingAccount1234@Banking" where Table.name = "customer_dim" and tableType = "external"
- Table hasTerm “savingAccount1234@Banking" select name orderby name desc
- Table hasTerm “savingAccount1234@Banking" limit 2
- Table hasTerm “savingAccount1234@Banking" or Table hasTerm "salesTerm@salesGlossary"
- Table hasTerm “savingAccount1234@Banking" and Table isA Dimension
- Table hasTerm “savingAccoun1234t@Banking" and db.name = “Sales" or (Table.qualifiedName like "customer")
- Table where Table hasTerm “savingAccount1234@Banking"
- Table where (name = "customer_dim" and Table hasTerm “savingAccount1234@Banking")
- Table hasTerm "savingAccount1234@Banking" select count() as terms

Examples of Relationship attributes filtering:

- Table where db.name = “Sales4321”
- Table where name = “customer_dim” select columns
- Table where columns.name like “sales” and Table isA Dimension
- Table where db.name = “Sales4321” limit 2
- Table where db.name = “Sales4321” orderby name asc
- Table where db.name = “Sales4321” and columns.name like “sales” and Table hasTerm “salesTerm@salesGlossary” - (Combination of both where and hasTerm attribute and keyword respectively.)

Related Information

Atlas Advanced Search language reference

Apache Atlas Advanced Search (atlas.apache.org)