Cloudera Runtime 7.1.9

# **Iceberg support for Atlas**

Date published: 2023-09-07 Date modified:



https://docs.cloudera.com/

## **Legal Notice**

© Cloudera Inc. 2024. All rights reserved.

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

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

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

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

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

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

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

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

## Contents

Iceberg support for Atlas	4
How Atlas works with Iceberg	
Using the Spark shell	
Using the Impala shell	

## **Iceberg support for Atlas**

Atlas integration with Iceberg helps you identify the Iceberg tables to scan data and provide lineage support. Learn how Atlas works with Iceberg and what schema evolution, partition specification, partition evolution are with examples.

### How Atlas works with Iceberg

You can use Atlas to find, organize, and manage different aspects of data about your Iceberg tables and how they relate to each other. This enables a range of data stewardship and regulatory compliance use cases.

The Atlas connectors distinguish between Hive and Iceberg tables. The Iceberg table is available in a typedef format which implies that the underlying data can be retrieved by querying the Iceberg table. All attributes of the Hive table are available in the Iceberg table and this equivalence is achieved by creating the Iceberg table as a sub-type of the underlying Hive table. Optionally, the Iceberg table can also be queried by Hive or Impala engine. For more information about Iceberg and related concepts, see Apache Iceberg features and Apache Iceberg in CDP.

Both Iceberg and Hive tables have equality in Atlas in terms of data tagging. Data evolution and transformation are features unique to Iceberg tables. Iceberg adds tables to compute engines including Spark, Hive and Impala using a high-performance table format that works just like a SQL table. Also, the lineage support for Iceberg table is available. For example, when a Hive table is converted to Iceberg format, the lineage is displayed for the conversion process in Atlas UI.



Attention: Whenever a classification is applied on Iceberg entities, Tag-based policies are supported for Iceberg entities.

• Migration of Hive tables to Iceberg is achieved with the following:

- Using in-place migration by running a Hive query with the ALTER TABLE statement and setting the table properties
- Executing CTAS command from Hive table to the Iceberg table.
- Schema evolution allows you to easily change a table's current schema to accommodate data that changes over time. Schema evolution enables you to update the schema that is used to write new data while maintaining backward compatibility with the schemas of your old data. Later the data can be read together assuming all of the data has one schema.
  - Iceberg tables supports the following schema evolution changes:

Add - add a new column to the table or to a nested struct

Drop- remove an existing column from the table or a nested struct

Rename- rename an existing column or field in a nested struct

Update- widen the type of a column, struct field, map key, map value, or list element

Reorder - change the order of columns or fields in a nested struct

• Partition specification allows you to initiate queries faster by grouping similar rows together when writing.

As an example, queries for log entries from a logs table usually include a time range, like the following query for logs between 10 A.M. and 12 A.M.

SELECT level, message FROM logs

WHERE event\_time BETWEEN '2018-12-01 10:00:00' AND '2018-12-0112:00:00'

Configuring the logs table to partition by the date of event\_time groups log events into files with the same event date. Iceberg keeps track of that date and uses it to skip files for other dates that do not have useful data.

• Partition evolution across Iceberg table partitioning can be updated in an existing table because queries do not reference partition values directly.

When you evolve a partition specification, the old data written with an earlier specification remains unchanged. New data is written using the new specification in a new layout. The metadata for each of the partition versions is stored separately.

Due to this nature of partition evolution, when you start writing queries, you get split planning. This is where each partition layout plans files separately using the filter it derives for that specific partition layout.

Related Information Using the Spark shell Using the Impala shell

### **Using the Spark shell**

Using Spark, you can create an Iceberg table followed by schema evolution, partition specification, and partition evolution.

#### Before you begin

You must configure the Spark shell as such you have included the valid Spark runtime version.

Run the following command in your Spark shell to create a new Iceberg table

#### **Procedure**

The following images provide information about Iceberg table creation process.

■ Sack To Results Q sample_1	× 1	lıl 🛔 admin
sample_1 (iceberg_tab         Classifications:         +         Terms:         +         Properties       Lineage         Relationships       Classifications	le) Audits Tasks	
✓ Technical properties	> User-defined properties	Add
columns (2) id v data	> Labels	Add
createTime 03/27/2023 10:38:28 AM (IST)	> Business Metadata	Add
db default@primary 💙		
■ ■ Back To Results Q sample_1	×	🔟 💄 admin
classifications:       +         Yroperties       Lineage       Relationships       Classifications       A	le)	
O Current Entity ∑ In Progress → Lineage → Impact		Q Q Q 4
/user/hive/wareho default.sample_1@ sam	pie_1	

Back To Results Q sample_1		🗙 🛄 🚢 admin
Classifications: +		
Terms: +		
Properties Lineage Relationships Classifications Audits	Tasks	
O Current Entity  ☐ In Progress → Lineage → Impact		
		iceberg_table
	guid	3b7b9d10-d773-4281-a318-7d30c6886c9f
	typeName	iceberg_table
	name	sample_1
/user/hive/wareho default.sample_1@ sample_1	qualifiedName	default.sample_1@primary
	owner	spark
	createTime	1679893708000
	status	ACTIVE
	classifications	N/A.

Run the following command in your Spark shell to create a Schema Evolution in a new table. For example - sample\_2.

**3.** spark.sql("CREATE TABLE spark\_catalog.default.sample\_2 ( id bigint COMMENT 'unique id', data string) USING iceberg");

The following image provide information about Iceberg schema evolution process.

Back To Result	Q sample_2	× 🛄 🏝 admin
📑 sa	mple_2 (iceberg_table)	
Classifications: 🕇		
Terms: +		
Properties I	ineage Relationships Classifications Audits Tasks	
raph 🔵 Table		
Key	Value	Show Empty Values
	Value data id	Show Empty Values
Key columns (2) db	data	

Run the following command in your Spark shell to include a column:

5. spark.sql("ALTER TABLE spark\_catalog.default.sample\_2 ADD COLUMN (add\_col\_1 string )");

The following images provide information about Iceberg schema creation process.

< Back To Re	Q Search entities			lıl.	🛔 admin
📑 sa	ample_2 (icel	perg_tab	le)		
assifications:	+				
erms: 🕇					
Properties	Lineage Relationships	Classifications A	udits Tasks		
<ul> <li>Technical p</li> </ul>	properties	• •	> User-defined properties		Add
columns (3)	data add_col_1 id	× v	> Labels		Add
createTime	03/27/2023 10:39:37 AM (IST)		> Business Metadata		Add
db	default@primary	~			
lastAccessTime	01/07/1970 05:29:24 PM (IST)				

Back To Results	<b>Q</b> Search entities		U.	lul 🛔 a
Users \$	Timestamp 🔻		Actions \$	
spark	03/27/2023 10:39:57 AM (IST)		Entity Updated	
me: sample_2				
✓ Technical prop	erties	✓ Relationsh	ip properties	
comment createTime	N/A 03/27/2023 10:39:37 AM (IST)	columns (3)	id data add col 1	× I
lastAccessTime	01/07/1970 05:29:24 PM (IST)			
name	sample_2	db	default	~
owner	spark			
parameters	<pre>{      v owner: "spark",</pre>	partitionKeys		N/A
	•	sd	default.sample_2@priman	y_stor
qualifiedName	default.sample_2@primary		5-	
retention	2147483647			
tableType	EXTERNAL_TABLE			

Run the following command in your Spark shell to include the second column:

7. spark.sql("ALTER TABLE spark\_catalog.default.sample\_2 ADD COLUMN (add\_col\_2 string )");

The following image provide information about Iceberg schema creation process.

<b>C</b> Search entities	<u>hi</u>	admii
sample_2 (iceberg_tab	le)	
Classifications: + Terms: +		
	Audits Tasks	
Users ≑ Timestamp ▼	Actions \$	
<ul> <li>spark 03/27/2023 10:40:27 AM (IST)</li> </ul>	Entity Updated	
Name: sample_2	✓ Relationship properties	
comment N/A	columns (4)	Ţ
createTime 03/27/2023 10:39:37 AM (IST)	data add col 1	
lastAccessTime 01/07/1970 05:29:24 PM (IST)	add_col_2	
name sample_2	db default 🗸	ה
	uerautt	

Run the following command in your Spark shell to create a Partition Specification in a new table (sample\_3):

**9.** spark.sql("CREATE TABLE spark\_catalog.default.sample\_3 (id bigint,data string,category string,ts timestamp) USING iceberg PARTITIONED BY (bucket(16, id), days(ts), category)");

The following images provide information about Iceberg partition specification process.

s s	ample_3 (ice	berg_tab	le)	
lassifications:	+			
erms: +				
Properties	Lineage Relationships	Classifications	Audits Tasks	
✓ Technical	properties		> User-defined properties	Add
columns (4)	ts id category	× •	> Labels	Add
createTime	03/27/2023 10:41:24 AM (IST)		> Business Metadata	Add
db	default@primary	~		
lastAccessTime	<ul> <li>01/07/1970 05:31:11 PM (IST)</li> </ul>			

createTime	03/27/2023 10:41:24 AM (IST)
db	default@primary 💙
lastAccessTime	01/07/1970 05:31:11 PM (IST)
name	sample_3
owner	spark
parameters	<pre>{     owner: "spark",     "current-schema": " </pre>
partitionSpec (3)	<pre>category, id_bucket, ts_day</pre>
qualifiedName	default.sample_3@primary
retention	2147483647
sd	default.sample_3@primary_stora

Run the following command in your Spark shell to create a Partition Evolution in a new table (sample\_3): **11.** spark.sql("ALTER TABLE spark\_catalog.default.sample\_3 ADD PARTITION FIELD years(ts)");

The following images provide information about Iceberg partition evolution process.

name	sample_3	
owner	spark	
parameters	<pre>{     owner: "spark",     previous_metadata_location: •</pre>	
partitionSpec (4)	<pre>category, ts_year, id_bucket, ts_day</pre>	
qualifiedName	default.sample_3@primary	
retention	2147483647	
sd	default.sample_3@primary_stora	
tableType	EXTERNAL_TABLE	
temporary	false	
typeName	iceberg_table	

Displaying Snapshot attributes

Run the following command to display relevant snapshot table parameters as attributes in Atlas. For example: Existing snapshot ID, current snapshot timestamp, snapshot count, number of files and related attributes.

spark.sql("CREATE TABLE spark\_catalog.default.sample3 ( id int, data string) USING iceberg");

spark.sql("INSERT INTO default.sample3 VALUES (1, 'TEST')");

Q sample3		×	<u>lılı</u>	🛔 admin
Entities				
/user/hive/	varehouse/ <b>sample3</b> (hdfs_path)			
sample3 (i	ceperg_table)			
default. <b>sar</b>	<pre>ple3@primary_storage (hive_storagedesc)</pre>			
Suggestions				
sample3				

lastAccessTime	06/19/2023 10:06:11 AM (IST)
name	sample3
numFiles	1
owner	spark
parameters	<pre>{     owner: "spark",     totalSize: "1226", </pre>
qualifiedName	l default.sample3@primary
retention	2147483647
sd	<pre>default.sample3@primary_storag</pre>
snapshotCount	0
tableType	EXTERNAL_TABLE
temporary	false

Run the following command to scale up the snapshot count value.

spark.sql("INSERT INTO default.sample3 VALUES (1, 'TEST')");

The latest Snapshot ID that Iceberg points to along with the Snapshot timestamp and Snapshot count are updated respectively.

=	Back To Results	Q Search entities		Laul	admin 🌡
	<ul> <li>recriment property</li> </ul>		<ul> <li>ose, defined propercies</li> </ul>		[
	columns (2)	id 💙 data	> Labels		Add
	createTime	06/19/2023 10:06:11 AM (IST)	> Business Metadata		Add
	currentSnapshotId	7368381093036867296			
	currentSnapshotTsMs	1687149412435			
	db	default.			
	lastAccessTime	06/19/2023 10:06:11 AM (IST)			
	name	sample3			
	numFiles	1			
	owner	spark			
=	Back To Results	Q Search entities		lahl	admin
	name	sample3			
	numFiles	1			
	owner	spark			
	parameters	<pre>{     v     owner: "spark",     previous_metadata_loca ▼</pre>			
	qualifiedName	default.sample3@primary			
	retention	2147483647			
	sd	default.sample3@primary_s			
	snapshotCount	1			
	tableType	EXTERNAL_TABLE			
	temporary	false			

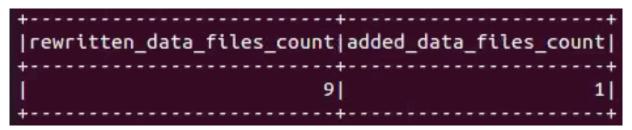
Click on the parameters field to display the details pertaining to the snapshot attributes.

Support for data compaction

Data Compaction is the process of taking several small files and rewriting them into fewer larger files to speed up queries. When performing compaction on an Iceberg table, execute the rewriteDataFiles procedure, optionally specifying a filter of which files to rewrite and the desired size of the resulting files. As an example, in an Atlas instance consider that the number of files and snapshot count are 9.

Run the following command to perform data compaction

spark.sql("CALL spark\_catalog.system.rewrite\_data\_files('default.sample3')"),show();



The count for the number of rewritten data files is compacted from a total count of 9 to 1.

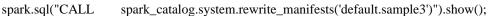
-

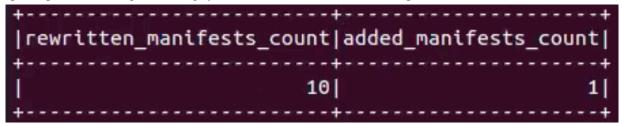
=	Back To Results	Q Search entíties
	currentSnapshotTsMs	1687149580886
	db	default
	lastAccessTime	06/19/2023 10:06:11 AM (IST)
	name	sample3
	numFiles	1
	owner	spark
	parameters	<pre>{     owner: "spark",     previous_metadata_loca •</pre>
	qualifiedName	default.sample3@primary
	retention	2147483647
	sd	default.sample3@primary_s
	snapshotCount	18 10

Support for metadata rewrite attributes

Iceberg uses metadata in its manifest list and manifest files speed up query planning and to prune unnecessary data files. You can rewrite manifests for a table to optimize the plan to scan the data.

Run the following command to rewrite manifests on a sample table 3.





The count for table manifested data is rewritten from 10 to 1.

Back To Results	Q Search entíties
lastAccessTime	06/19/2023 10:06:11 AM (IST)
manifestsCreated	1
manifestsKept	0
manifestsReplaced	10
name	sample3
numFiles	1
owner	spark

The process is completed.

Related Information Using the Impala shell

### Using the Impala shell

Using Impala, you can create an Iceberg table followed by Schema evolution, partition specification, partition evolution and CTAS operation.

#### Before you begin

Run the following command in your Impala shell to create a new Iceberg table

#### **Procedure**

1. CREATE TABLE ice\_t (i INT) STORED AS ICEBERG;

2. Navigate accordingly in the Atlas UI to view the changes.

The following images provide information about Iceberg table creation process.

=	Back To Results	Q ice_t	×	<mark>.lıl</mark>	🛔 admin
(	ice_t	t (iceberg_table)			
Clas	ssifications: +				
Terr	ms: 🛨				
	Propenties Lines	ge Relationships Classifications Audits Tasks			
00	urrent Entity 🛛 In Pro	gress → Lineage → Impact	13 🙆 🗘 1	r Q	Q Q 4
	/user/hive/wareho	default.ice_t@pri ice_t			

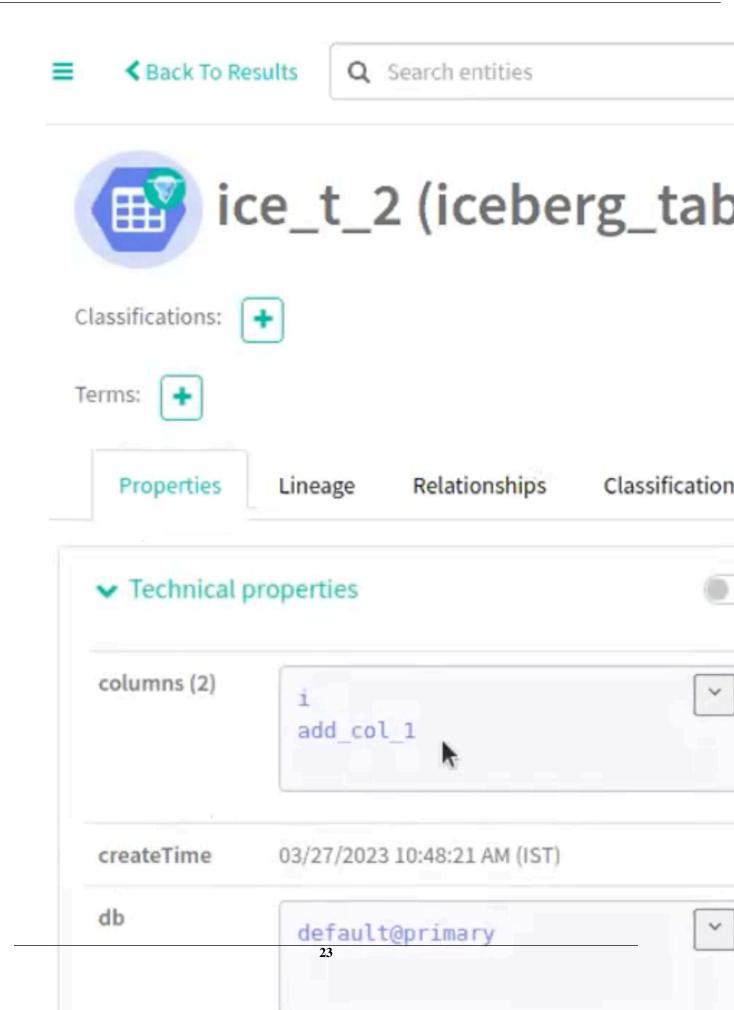
Run the following command in your Impala shell to create a scheme evolution:

**3.** CREATE TABLE ice\_t\_2 (i INT) STORED AS ICEBERG;

< Back To Re	Q ice_t_2			×	ht	🛔 admin
ic 🕑	ce_t_2 (iceberg	_table)				
lassifications:	+					
erms: +						
Properties	Lineage Relationships Cl	lassifications	Audits Tasks			
✓ Technical p	properties		> User-defined properties			Add
<ul> <li>Technical p</li> <li>columns (1)</li> </ul>	properties 1		<ul> <li>&gt; User-defined properties</li> <li>&gt; Labels</li> </ul>			Add

Run the following command in your Impala shell to add a column to the existing table (ice\_t\_2):

4. alter table ice\_t\_2 ADD COLUMNS (add\_col\_1 string );



Run the following command in your Impala shell to create a partition specification.

5. CREATE TABLE ice\_part\_spec (s string , b string ) PARTITIONED BY SPEC (truncate(3, s)) STORED AS ICEBERG ;

Back To Res	ults Q ice_part_spec		🗙 🛄 🏝 admin
ic 🚯	e_part_spec (iceberg_	table)	
assifications:	•		
erms: +	i e all		
<u> </u>			
Properlies	Lineage Relationships Classifications Au	idits Tasks	
Users \$	Timestamp *	Actio	ons \$
impala	03/27/2023 10:49:28 AM (IST)	Entity	Created
wing <u>1 records</u> Fr	om 1 - 25		Page Limit : 25
Back To Res	ults Q ice_part_spec		🗙 📶 🖀 admin
lastAccessTime	01/07/1970 05:39:15 PM (IST)		
name	ice_part_spec		
owner	impala		
parameters	<pre>{     "external.table.purge":     "TRUE",     "</pre>		
partitionSpec (1)	s_trunc		
qualifiedName	default.ice_part_spec@primary		
retention	2147483647		
sd	<pre>default.ice_part_spec@primary_ orage</pre>	•	
tableType	EXTERNAL_TABLE		
temporary	false		
typeName	iceberg_table		

Run the following command in your Impala shell to create a partition evolution.

**6.** CREATE TABLE ice\_p (i INT, d DATE, s STRING, t TIMESTAMP) PARTITIONED BY SPEC (BUCKET(5, i), MONTH(d), TRUNCATE(3, s), HOUR(t))STORED AS ICEBERG;

Back To Res	ults Q ice_p	× III	admi admi
lastAccessTime	01/07/1970 05:40:00 PM (IST)		
name	ice_p		
owner	impala		
parameters	<pre>{     "external.table.purge":     "TRUE",     "</pre>		
partitionSpec (4)	<pre>s_trunc, t_hour, i_bucket, d_month</pre>		
qualifiedName	default. <mark>ice_p</mark> @primary		
retention	2147483647		
sd	default.ice_p@primary_storage		
tableType	EXTERNAL_TABLE		

Run the following command in your Impala shell to modify the partition specification

7. ALTER TABLE ice\_p SET PARTITION SPEC (VOID(i), VOID(d), TRUNCATE(3, s), HOUR(t), i);

< Back To	Results Q Search entities
lastAccessT	me 01/07/1970 05:40:00 PM (IST)
name	ice_p
owner	impala
parameters	<pre>{     previous_metadata_location:     "hdfs://atlas- </pre>
partitionSp (5)	c d_null, s_trunc, t_hour, i_nul
qualifiedNa	ne default.ice_p@primary
retention	2147483647
sd	default.ice_p@primary_storage
	27

Run the following commands in your Impala shell to create the contents of one table (ice\_t\_3) to another table (ice\_t\_4).

**8.** CREATE TABLE ice\_t\_3 (i INT) STORED AS ICEBERG;

**9.** CREATE TABLE ice\_t\_4 STORED AS ICEBERG as select \* from ice\_t\_3;

Back To Results	Q ice_t_4
	Entities
	/user/hive/warehouse/ice_t_4
partitionSpec (5)	default.ice_t_4@primary:-100
	ice_t_4 (iceberg_table)
	default.ice_t_4@primary_stor
	default.ice_t_4@primary:1679
impala	Suggestions
	ice_t_4
ie: ice_p	
<ul> <li>Technical property</li> </ul>	erties
comment	N,
createTime	03/27/2023 10:50:13 AM (IS
lastAccessTime	01/07/1970 05:40:00 PM (IS
name	ice_
	partitionSpec (5) impala ne: ice_p Technical prope comment createTime lastAccessTime

