

Managing Virtual Warehouses

Date published: 2024-01-01

Date modified: 2024-12-05



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

Virtual Warehouse IP address and cloud resource requirements for on cloud environments.....	5
IP address and cloud resource requirements for Virtual Warehouses running on AWS environments.....	5
IP address and cloud resource requirements for Virtual Warehouses running on Azure environments.....	7
Creating a Virtual Warehouse.....	9
Compute Instance Types.....	10
Virtual Warehouse sizing requirements for public cloud environments.....	11
How to size a Virtual Warehouse.....	14
Auto-suspend timeout in Cloudera Data Warehouse.....	14
Configuring auto-scaling.....	15
Configuring Hive VW concurrency auto-scaling.....	15
Configuring Hive query isolation auto-scaling.....	16
Configuring Impala Virtual Warehouse auto-scaling.....	17
Configuring Workload Aware Autoscaling.....	18
Configuring Impala coordinator shutdown.....	20
Configuring Impala coordinator high availability.....	22
Configuring Impala catalog high availability.....	25
Configuring Impala Statestore high availability.....	26
Impala JDBC/ODBC host names.....	26
Creating a Virtual Warehouse on ARM instances.....	27
Configuring a Virtual Warehouse.....	27
Tuning Virtual Warehouses.....	28
Tuning Impala auto-scaling.....	28
Tuning Impala Java tool options.....	29
Disabling metadata synchronization.....	29
Disabling Impala catalog high availability.....	30
Enabling SSO to a Virtual Warehouse.....	30
Configuring token-based authentication.....	32
Generating a JWT token from the Cloudera side.....	32
Configuring the Impala Virtual Warehouse for authentication.....	35
Configuring Impyla for authentication.....	35
Configuring the Hive Virtual Warehouse for authentication.....	36
Getting a JDBC URL for your Virtual Warehouse.....	37
Authenticating from a JDBC client.....	37
Connecting Cloudera Data Warehouse and Cloudera AI.....	38
Configuring a time zone.....	38
Configuring UDF JAR caching in Hive Virtual Warehouse.....	39
Connecting Cloudera Data Warehouse and Kudu.....	40
Gathering Kudu master FQDNs.....	40
Configuring a Cloudera Data Warehouse-Kudu connection.....	42
Setting up a past release for a Cloudera Data Warehouse-Kudu connection.....	43
Configuring Iceberg manifest caching in Impala Virtual Warehouse.....	45
Managing high partition workloads.....	47
Configuring an Operational Database connection.....	48
Setting up Cloudera Data Warehouse to connect to Cloudera Operational Database.....	48
Resizing the Impala Virtual Warehouse size.....	49
Resizing the Hive Virtual Warehouse size.....	49
Compaction in Cloudera Data Warehouse.....	50
Compaction prerequisites.....	50
Change compactor configuration for Hive Virtual Warehouses on Cloudera Data Warehouse on cloud.....	51

Initiating automatic compaction.....	51
Configuring compaction.....	52
Automating metadata invalidation after compaction.....	52
How compaction works.....	53
Compactor processes.....	53
How compaction interacts with the Data Lake.....	53
Cloudera Data Warehouse on cloud Compaction Architecture.....	54
Compaction observability.....	54
Viewing a compaction alert using Grafana.....	55
Adding a compaction alert.....	56
Managing a compaction alert.....	57
Configuring alert notifications.....	58
Configuring alert notifications, no UI.....	59
Checking an alert notification configuration.....	61
Deleting a compaction alert.....	62
Data Visualization in Cloudera Data Warehouse.....	62
Rebuilding a Database Catalog.....	63
Backing up and restoring Hue documents.....	64
Rebuilding a Virtual Warehouse.....	65
Upgrading Database Catalogs and Virtual Warehouses in Cloudera Data Warehouse on cloud.....	66
Debugging with Impala Web UIs.....	68

Virtual Warehouse IP address and cloud resource requirements for on cloud environments

Learn about how many IP addresses and cloud resources are required to run Virtual Warehouses efficiently in public cloud environments for Cloudera Data Warehouse on cloud.

IP address and cloud resource requirements for Virtual Warehouses running on AWS environments

Learn about the estimated number of IP addresses and cloud resources required to run Virtual Warehouses on AWS environments for Cloudera Data Warehouse on cloud.



Important: These requirements are estimated. Your particular workloads and configurations can affect the number of IP addresses and cloud resources required to run Virtual Warehouses efficiently. In addition, these requirements are for AWS environments that use the default AWS VPC Container Networking Interface (CNI) plugin. To reduce the number of required IP addresses, you can enable the overlay networking feature for AWS environments in . For further details, see the link to "Overlay networks for AWS environments in " at the bottom of this page.

Virtual Warehouse requirements:

Each compute node in a Virtual Warehouse that runs on AWS environments requires 8 IP addresses. Each executor needs one compute node, so the size of your Virtual Warehouse contributes to the number of IP addresses required. To calculate the number of IP addresses required for custom sizes, multiply the number of executors by 8 and add for the shared services as specified in the following sections.

Shared services requirements for Database Catalogs:

Virtual Warehouses also require shared services for the Database Catalog.

Additional shared services requirements:

The usage for other shared services is different for Hive Virtual Warehouses and Impala Virtual Warehouses. Here are the different requirements for Hive versus Impala Virtual Warehouses:

- Hive Virtual Warehouses add 1 compute node for each executor and 1 shared services node for the HiveServer for each Virtual Warehouse.
- Impala Virtual Warehouses add 1 compute node for each executor, 1 or 2 compute nodes for the coordinator, depending on the HA configuration, and 1 shared services node per Virtual Warehouse for Impala catalogd.

The following tables summarize the approximate number of IP addresses and cloud resources you should plan for Virtual Warehouses on AWS environments.

Table 1: Hive Virtual Warehouses running on AWS environments

Size (# executors)	# Compute nodes for executors	# Shared services nodes Database Catalog	# Shared services nodes for HiveServer	Total IP addresses required
XSMALL (2)	2	3	1	Executor nodes: 2 nodes X 8 = 16 Shared services nodes: 3 + 1 = 4 nodes X 25 = 100 TOTAL = ~116 IP addresses

Size (# executors)	# Compute nodes for executors	# Shared services nodes Database Catalog	# Shared services nodes for HiveServer	Total IP addresses required
SMALL (10)	10	3	1	Executor nodes: 10 nodes X 8 = 80 Shared services nodes: 3 + 1 = 4 nodes X 25 = 100 TOTAL = 180 IP addresses
MED (20)	20	3	1	Executor nodes: 20 nodes X 8 = 160 Shared services nodes: 3 + 1 = 4 nodes X 25 = 100 TOTAL = 260 IP addresses
LARGE (40)	40	3	1	Executor nodes: 40 nodes X 8 = 320 Shared services nodes: 3 + 1 = 4 nodes X 25 = 100 TOTAL = 420 IP addresses

Table 2: Impala Virtual Warehouses running on AWS environments

Size (# executors)	# Compute nodes for executors	# Compute nodes for coordinator	# Shared services nodes for Impala catalogd	# Shared services nodes for Database Catalog	Total IP addresses required
XSMALL (2)	2	1-2	1	3	Executor/ coordinator nodes: 3-4 nodes X 8 = 24-32 Shared services nodes: 1 + 3 = 4 nodes X 25 = 100 TOTAL = 124 to 132 IP addresses
SMALL (10)	10	1-2	1	3	Executor/ coordinator nodes: 11-12 nodes X 8 = 88-96 Shared services nodes: 1 + 3 = 4 nodes X 25 = 100 TOTAL = 188 to 196 IP addresses
MED (20)	20	1-2	1	3	Executor/ coordinator nodes: 21-22 nodes X 8 = 168-176 Shared services nodes: 1 + 3 = 4 nodes X 25 = 100 TOTAL = 168 to 176 IP addresses
LARGE (40)	40	1-2	1	3	Executor/ coordinator nodes: 41-42 nodes X 8 = 328-336 Shared services nodes: 1 + 3 = 4 nodes X 25 = 100 TOTAL = 428 to 436 IP addresses

Related Information[Overlay networks for AWS environments in Cloudera Data Warehouse](#)

IP address and cloud resource requirements for Virtual Warehouses running on Azure environments

Learn about the estimated number of IP addresses and cloud resources required to run Virtual Warehouses on Azure environments for Cloudera Data Warehouse on cloud.



Important: These requirements are estimated. Your particular workloads and configurations can affect the number of IP addresses and cloud resources required to run Virtual Warehouses efficiently. In addition, these requirements are for Azure environments that use the default Azure Container Networking Interface (CNI) plugin. To reduce the number of required IP addresses, you can enable the kubenet networking feature for Azure environments in .

Virtual Warehouse requirements:

Each compute node in a Virtual Warehouse that runs on Azure environments requires 16 IP addresses (for pods and for the node itself). Each executor needs one compute node, so the size of your Virtual Warehouse contributes to the number of IP addresses required. To calculate the number of IP addresses required for custom sizes, multiply the number of executors by 16 and add for the shared services as specified in the following sections.

Shared services requirements for Database Catalogs:

Virtual Warehouses also require shared services for Database Catalogs.

Additional shared services requirements:

The usage for other shared services is different for Hive Virtual Warehouses and Impala Virtual Warehouses. Here are the different requirements for Hive versus Impala Virtual Warehouses:

- Hive Virtual Warehouses add 1 compute node for each executor and 1 shared services node for the HiveServer for each Virtual Warehouse.
- Impala Virtual Warehouses add 1 compute node for each executor, 1 or 2 compute nodes for the coordinator, depending on the HA configuration, and 1 shared services node per Virtual Warehouse for Impala catalogd.

The following tables summarize the approximate number of IP addresses and cloud resources you should plan for Virtual Warehouses on Azure environments.

Table 3: Hive Virtual Warehouses running on Azure environments

Size (# executors)	# Compute nodes for executors	# Shared services nodes for Database Catalog	# Shared services nodes for HiveServer	Total IP addresses required
XSMALL (2)	2	3	1	Executor nodes: $2 \text{ nodes} \times 16 = 32$ Shared services nodes: $3 + 1 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 156 IP addresses
SMALL (10)	10	3	1	Executor nodes: $10 \text{ nodes} \times 16 = 160$ Shared services nodes: $3 + 1 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 284 IP addresses

Size (# executors)	# Compute nodes for executors	# Shared services nodes for Database Catalog	# Shared services nodes for HiveServer	Total IP addresses required
MED (20)	20	3	1	Executor nodes: $20 \text{ nodes} \times 16 = 320$ Shared services nodes: $3 + 1 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 444 IP addresses
LARGE (40)	40	3	1	Executor nodes: $40 \text{ nodes} \times 16 = 640$ Shared services nodes: $3 + 1 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 764 IP addresses

Table 4: Impala Virtual Warehouses running on Azure environments

Size (# executors)	# Compute nodes for executors	# Compute nodes for coordinator	# Shared services nodes for Impala catalogd	# Shared services nodes for Database Catalog	Total IP addresses required
XSMALL (2)	2	1-2	1	3	Executor/ coordinator nodes: $3-4 \text{ nodes} \times 16 = 48-64$ Shared services nodes: $1 + 3 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 172 to 188 IP addresses
SMALL (10)	10	1-2	1	3	Executor/ coordinator nodes: $11-12 \text{ nodes} \times 16 = 176-192$ Shared services nodes: $1 + 3 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 300 to 316 IP addresses
MED (20)	20	1-2	1	3	Executor/ coordinator nodes: $21-22 \text{ nodes} \times 16 = 336-352$ Shared services nodes: $1 + 3 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 460 to 476 IP addresses
LARGE (40)	40	1-2	1	3	Executor/ coordinator nodes: $41-42 \text{ nodes} \times 16 = 656-672$ Shared services nodes: $1 + 3 = 4 \text{ nodes} \times 31 = 124$ TOTAL = 780 to 796 IP addresses

Related Information
[Overlay networks for Azure environments in Cloudera Data Warehouse](#)

Creating a Virtual Warehouse

A Virtual Warehouse is an instance of compute resources in on cloud that is equivalent to an on-prem cluster. You learn how to create a new Virtual Warehouse in Cloudera Data Warehouse on cloud.

About this task

A Virtual Warehouse provides access to the data in tables and views in the data lake your Database Catalog uses. A Virtual Warehouse can access only the Database Catalog you select during creation of the Virtual Warehouse.

In this task and subtasks, you configure Virtual Warehouse features, including performance-related features for production workloads, such as the Virtual Warehouse size and auto-scaling. These features are designed to manage huge workloads in production, so if you are evaluating Cloudera Data Warehouse, or just learning, simply accept the default values. This task covers the bare minimum configurations.

In subtopics, you see details about how to configure features for production workloads, such as Hive query isolation and Impala catalog high availability.

Before you begin

- You obtained permissions to access a running environment for creating a Virtual Warehouse.
- You obtained the DWAdmin role to perform Cloudera Data Warehouse tasks.
- You logged into the Cloudera web interface.
- You activated the environment from Cloudera Data Warehouse.

For more information about meeting these prerequisites, see [Getting started in Cloudera Data Warehouse](#).

Procedure

1. Navigate to Data Warehouses Virtual Warehouses Create Virtual Warehouse .
2. In New Virtual Warehouse, specify a Name.



Note: The fully qualified domain name of your Virtual Warehouse, which includes the Virtual Warehouse name plus the environment name must not exceed 64 characters; otherwise, Hue cannot load.

3. Select the Hive or Impala type of Virtual Warehouse you want.

Virtual Warehouses can use Hive or Impala as the underlying SQL execution engine. Typically, Hive is used to support complex reports and enterprise dashboards. Impala is used to support interactive, ad-hoc analysis.

4. Select the Environment and Database Catalog for this Virtual Warehouse.
5. In AWS environments only, accept the default availability zone, or select an availability zone, such as us-east-1c.
The default behavior is to randomly select an availability zone from the list of configured availability zones for the associated environment. Generally, it is fine to accept the default. All compute resources will run in the selected zone.



Note: Selection of the zone is not an option in Azure environments.


6. Select the Compute Instance Types for the Virtual Warehouse based on your workload.
For more information, see *Supported Compute Instance Types*.
7. Select the Hive Image Version or the Impala Image Version version, and the Hue Image version you want to use, or accept the default version (latest) at the top of the drop-down menus.
8. Select the Size of the Virtual Warehouse as described in the next subtopic.
The “AutoSuspend Timeout” and “Concurrency Autoscaling” controls appear.
9. Configure auto-scaling as described in the subsequent subtopic.

10. In Authentication, select Enable SSO to enable [single sign-on to your Virtual Warehouse](#), and in User Groups, select a user group set up in advance to access endpoints. If you do not have a user group set up for SSO, do not select Enable SSO.

In Management Console User Management you set up a user group, required for enabling SSO, that identifies the users authorized to access to this Virtual Warehouse.

11. Enter keys and values for Tagging the Virtual Warehouse.

12. Accept default values for other settings, or change the values to suit your use case, and click Create Virtual Warehouse to create the new Virtual Warehouse.

Click the tooltip  for information about settings.

When you create a Virtual Warehouse, a cluster is created in your cloud provider account. This cluster has two buckets. One bucket is used for managed data and the other is used for external data.

Related Information

[Supported compute instance types](#)

[Custom tags in AWS environments](#)

[Custom tags in Azure environments](#)

Compute Instance Types

Learn about the supported AWS and Azure compute instance types that you can select while creating a Virtual Warehouse.

Supported AWS compute instance types

Cloudera Data Warehouse supports the following AWS compute instance types (Hive and Impala executors):

Instance type	Processor	Usage	Virtual Warehouse Support
r7gd.4xlarge	ARM	Compute	Impala
r6gd.4xlarge	ARM	Compute	Impala
r6id.4xlarge	Intel	Compute	Hive and Impala
r5d.4xlarge	Intel	Compute (default)	Hive and Impala
r5ad.4xlarge	AMD	Compute	Hive and Impala
r5dn.4xlarge	Intel	Compute	Hive and Impala
m5.2xlarge	Intel	Shared services	Hive and Impala

In the Cloudera Data Warehouse environment, instances for shared service components are set up within a Kubernetes (K8s) cluster. The setup begins with three m5.2xlarge instances running the Cloudera Data Warehouse service, but the K8s cluster is capable of autoscaling, automatically adding more instances if necessary to handle increased demand. Additionally, an Amazon Relational Database Service (RDS) (db.r5.large) running PostgreSQL is created to store user metadata for Hue and Data Visualization services. In total, three shared db.r5.large nodes are used for this purpose. For more information, see [Always active, shared services](#).

Supported Azure compute instance types

Cloudera Data Warehouse supports the following Azure compute instance types (Hive and Impala executors):

Azure VM	Processor Type	Usage	Virtual Warehouse Support
Standard_E16pds_v5	ARM	Compute	Impala
Standard_E16_v3	Intel	Compute	Hive and Impala
Standard_E16ds_v4	Intel	Compute (default)	Hive and Impala

Azure VM	Processor Type	Usage	Virtual Warehouse Support
Standard_E16ads_v5	AMD	Compute	Hive and Impala
Standard_E16ds_v5	Intel	Compute	Hive and Impala
Standard_D8s_v4	intel	Shared services (default)	Hive and Impala
Standard_D8as_v5	Intel	Shared services, used with AMD compute instance Standard_E16ads_v5	Hive and Impala

Three instances are added to the cluster as needed for shared services (always on components). Three shared nodes are dStandard_E2s_v3 MemoryOptimized using flexserver, for the Amazon Relational Database Service (RDS). These shared nodes are used for Hue and Data Visualization user metadata. For more information, see [Always active, shared services](#).

Virtual Warehouse sizing requirements for public cloud environments

You need to understand how to estimate size requirements for Cloudera Data Warehouse on cloud Virtual Warehouses to create a Virtual Warehouse that meets your needs.


Calculating requirements for on-premises data warehouse deployments

Selecting the correct size of environment before you migrate your workloads from CDH and HDP to is critical for preserving performance characteristics. Consider the following workload characteristics when you plan for capacity on your environment:

- Query memory requirements
- CPU utilizations
- Disk bandwidth
- Working set size
- Concurrent query execution requirements

As part of this calculation, it is important to understand the core hardware difference between and on-premises hosts as explained in the following table:

Table 5: Hardware differences between and on-premises hosts

Hardware component	On-premises	AWS R5D.4xlarge instance	Azure E16 v3 instance
CPU cores	20 - 80	16	16
Memory	128 GB minimum 256 GB+ recommended	128 GB	128 GB
Network	10 Gbps minimum 40 Gbps recommended	Up to 10 Gbps	Up to 8,000 Mbps
Instance storage	12 x 2 TB drives (1,000 MB/s sequential)	2 x 300 GB NVMe SSD (1,100 MB/s sequential)  Note: Optimally EBS volumes can be added to scratch and spool space for Impala.	400 GiB SSD
Persistent storage performance	At least 500 MB/s per disk With 20 disks, 10 GB/s per node	1,156 MB/s per EC2 instance	Maximum IOPS: 24,000 Maximum Read: 375 MB/s Maximum Write: 187 MB/s

An AWS R5D.4xlarge instance closely matches the CPU, memory, and bandwidth specifications that are recommended for CDH clusters. AWS R5D.4xlarge instance specifications are the default "instance type" for . AWS EBS storage cannot be used as primary database storage because it is transient and lacks sufficient capacity. This core difference makes it necessary to use a different strategy for than you used for CDH to achieve good scan performance.

Supported AWS instances

You can choose an instance type other than the default AWS R5D.4xlarge , and also a fallback instance type, using the CDP CLI. To fetch a string of available instance types, use the following CLI command:

```
cdp dw describe-allowed-instance-types
```

Example output:

```
{
  "aws": {
    "default": [
      "r5d.4xlarge"
    ],
    "allowed": [
      "r5d.4xlarge",
      ...
    ]
  }
}
```

sizing and scaling

Before you migrate data to , plan for scaling and concurrency. In the cloud, scaling and concurrency can elastically respond to workload demands, which enables the system to operate at a lower cost than you might expect. If you configure your environment to accommodate peak workloads as a constant default configuration, you may waste resources and money when system demand falls below that level.

In , the size of the Virtual Warehouse determines the number of executor instances for an individual cluster so it determines memory limits and performance capabilities of individual queries:

Table 6: Virtual Warehouse sizes

Size	Number of executors
X-Small	2
Small	10
Medium	20
Large	40

Warehouse size in combination with auto-scaling settings determine how many clusters are allocated to support concurrent query execution.

The Virtual Warehouse size must be at least large enough to support the memory used by the most data-intensive query. Usually, the Virtual Warehouse size does not need to be larger than the most data-intensive query. Better caching is provided if there is commonality between data sets accessed by queries. Increasing the Virtual Warehouse size can increase single-user and multi-user capacity because additional memory and resources allows larger datasets to be processed. Concurrent query execution is also supported by sharing resources. If too small a size is configured for the Virtual Warehouse, poor data caching and memory paging can result. If too large a size is configured, excessive public cloud costs are incurred due to idle executors.

The primary difference between and CDH on-premises deployments when choosing a warehouse size based on existing hardware:

- With all resources on an executor are dedicated to query processing.

- With CDH on-premises deployments resources support other operations in addition to query processing. For example, these on-premises resources are shared with other services, such as HDFS or other locally hosted file systems. In particular Spark, HBase, or MapReduce. These other services might consume significant resources.

Consequently, you might be able to choose a much smaller Virtual Warehouse size in because resources are isolated in their own pod in the environment.

In the case of Impala, it is useful to look at the per-process metrics to isolate the impalad backend and the Impala front-end Java processes that hold the Catalog cache. In , the Impala coordinator and executor roles are separated leaving the unused Catalog JVM memory free to support query execution. You should look at the memory utilization metrics for executor-only impalad nodes (those not also running the coordinator role) to estimate how much memory your current cluster of Impala executors requires.

Concurrency

Concurrency is the number of queries that can be run at the same time. Determine the size you need by considering the amount of resources your system needs to support peak concurrency.

By default, Impala Virtual Warehouses can run 3 large queries per executor group. Executors can handle more queries that are simpler and that do not utilize concurrency on the executor. For most read-only queries the default setting of 3 queries per executor group is sufficient. Occasional peaks are handled transparently by the auto-scaling feature. When auto-scaling is triggered an additional executor group is added thereby doubling query concurrency capacity. Scaling the Virtual Warehouse by adding more clusters enables additional concurrent queries to run, but does not improve single-user capacity or performance. Concurrently executed queries are routed to the different clusters and execute independently. The number of clusters can be changed to match concurrent usage by changing the auto-scaling parameters. For more details about auto-scaling settings, see the links at the bottom of this page.



Important: The number of very large queries that can be run might also be impacted by memory limits on the executors.

For Hive on LLAP Virtual Warehouses, each size setting indicates the number of concurrent queries that can be run. For example, an X-Small Hive on LLAP Virtual Warehouse can run 2 concurrent queries. A Small Virtual Warehouse can run 10 concurrent queries. To run 20 concurrent queries in a Hive on LLAP Virtual Warehouse choose Medium size.

Caching "Hot Datasets"

Frequently accessed data is sometimes referred to as a "hot dataset." CDH supports caching mechanisms on the compute nodes to cache the working set that is read from remote file systems, such as remote HDFS data nodes, S3, ABFS, or ADLS. This offsets the input/output performance difference.

In , frequently accessed data is cached in a storage layer on SSD so that it can be quickly retrieved for subsequent queries. This boosts performance. Each executor can have up to 200 GB of cache. For example, a Medium-sized Virtual Warehouse can keep $200 * 20 = 4$ TB of data in its cache. For columnar formats, such as ORC, data in the cache is decompressed, but not decoded. If the expected size of the hot dataset is 6 TB, which requires about 30 executors, you can over-provision by choosing a Large-sized warehouse to ensure full cache coverage. A case can also be made to under-provision by choosing a Medium-sized warehouse to reduce costs at the expense of having a lower cache hit rate. To offset this, keep in mind that columnar formats allow optimizations such as column projection and predicate push-down, which can significantly reduce cache requirements. In these cases, under-provisioning the cache might have no negative performance effect.

Scanned dataset size

Scanning large datasets on Amazon S3 or Azure ADLS can be slow. A single R5D.4xlarge EC2 instance can only scan data at 1,156 MB/s maximum throughput according to [standard S3 benchmarks](#). If a query must read 100 GB from S3, S3 scanning takes a minimum of 88 seconds on just one node. Depending on the number of files in the S3 directory, it might take more than 3 minutes. In this case, if a query needs to scan 100 GB of data, if you use 10 nodes, you can get the scan time down to approximately 20 seconds.

However, keep in mind that with columnar storage, minimum/maximum statistics in files, and other conditions, often the amount of data read is significantly less than the total size of the files for certain queries.

Related Tasks

[Tuning Impala auto-scaling](#)

Related Information

[Tuning Virtual Warehouses](#)

How to size a Virtual Warehouse

When you create a Virtual Warehouse, you need to carefully set the size of your Virtual Warehouse. The size of the Virtual Warehouse you select during Virtual Warehouse creation determines the number of executors and concurrent queries the Virtual Warehouse can run.

Before creating a Virtual Warehouse and setting the Virtual Warehouse size, learn about critical ["Warehouse sizing requirements"](#). If you misconfigure the size of your Virtual Warehouse, you can use one of the following methods, depending on the Impala or Hive type of Virtual Warehouse, to correct the size:

- Impala: [Edit your Virtual Warehouse configuration to correct the size.](#)
- Hive: [Delete and recreate the Virtual Warehouse to correct the size.](#)

When you create a Virtual Warehouse, you select one of the following Virtual Warehouse sizes:

Virtual Warehouse Size	Number of Executors
XSMALL	2
SMALL	10
MEDIUM	20
LARGE	40
Custom	Enter a value between '1' and '100'

If you are evaluating Cloudera Data Warehouse, or just learning, XSMALL is the recommended size. For production workloads, choose a size based on the following factors:

- The number of executors you typically use for clusters in an on-premises deployment.
- The complexity of your queries and the size of the data sets that they access.

Large warehouses with more executors can cache more data than small warehouses. Caching enhances performance.

Auto-suspend timeout in Cloudera Data Warehouse

Auto-suspend enables you to handle resources when the auto-scaler has scaled back to the last executor group. You can control the time that the original warehouse executor group idles after all other groups scale down and release their executors. The JDBC endpoint lives on to respond to queries from the result cache or statistics, but expensive executors no longer run.

When the auto-scaler increases or decreases the number of executor groups in the Virtual Warehouse, it can take several minutes before the scaling up or down takes effect. This slight delay is caused by the time required by your cloud provider to provision clusters.

When no queries are sent to an executor group, resources scale down and executors are released. When all executor groups are scaled back, when executors are idle, and after a period of idle time (Auto-suspend Timeout), the Virtual Warehouse is suspended.

You set an Auto-suspend Timeout to configure how long a Virtual Warehouse idles before shutting down.

Auto-suspend timeout is independent of the auto-scaling process and only applies to the original Virtual Warehouse and not to any additional warehouses that are created as a result of auto-scaling.

If a query is executed on a suspended Virtual Warehouse, then the query coordinator queues the query. When a queued query is detected, an executor group is immediately added (a scale-up occurs) to run the query on the Virtual Warehouse.

This feature can help lower cloud costs.

Auto-suspend differs from manually stopping a Virtual Warehouse. Sending a query to a stopped Virtual Warehouse returns an HTTP 503 "no healthy upstream" error, no scale-up occurs, and the query does not execute.

Auto-suspend is enabled by default.

While creating a Virtual Warehouse, set Auto-suspend Timeout to the number of seconds you want the Virtual Warehouse to idle before shutting down. For example, if you set the timeout to 300 seconds, the Virtual Warehouse suspends itself after 300 seconds to save compute resource expense. The suspended Virtual Warehouse restarts as soon as you run a query.

To disable auto-suspend, select the Disable Auto-suspend option. You can disable the auto-suspend option both while creating a Virtual Warehouse or after creating it.

Configuring auto-scaling

When you create a Virtual Warehouse, you set auto-scaling to increase and decrease resources according to demand. Auto-scaling relinquishes resources when demand decreases to limit unnecessary cloud expenses. Auto-scaling increases resources to speed performance.

Auto-scaling is designed for huge workloads in production, so if you are evaluating Cloudera Data Warehouse, or just learning, simply accept the default values. Later, you can edit the Virtual Warehouse to tune auto-scaling.

Before configuring or tuning auto-scaling, you need to understand the [auto-scaling process](#) to make good configuration decisions. As a Hive Virtual Warehouse user, choose the method of auto-scaling and see the relevant configuration subtopic (below):

- Concurrency for BI-type queries
- Isolation for ETL-type queries

To configure auto-scaling in a Hive or Impala Virtual Warehouse, see the topics below:

Configuring Hive VW concurrency auto-scaling

Configuring the Hive Virtual Warehouse to use concurrency auto-scaling is critical for controlling cloud expenses.

Before you begin

- You are familiar with the [auto-scaling process](#).
- You are creating a Hive Virtual Warehouse for running BI-type queries.
- In Cloudera Data Warehouse, you added a Hive Virtual Warehouse, configured the size of the Hive Virtual Warehouse, and configured auto-suspend as described in previous topics.
- You obtained the DWAdmin role.

About this task

In this task, first you select the number of executors for your virtual cluster.

- Minimum executors

The fewest executors you think you will need to provide sufficient resources for your queries. The number of executors needed for a cloud workload is analogous to the number of nodes needed for an on-premises workload.

- Maximum number

The maximum number of executors you will need. This setting limits prevents running an infinite number of executors and runaway costs.

Consider the following factors when configuring the minimum and maximum number of executors:

- Number of concurrent queries
- Complexity of queries
- Amount of data scanned by the queries
- Number of queries

Next, you configure when your cluster should scale up and down based on one of the following factors:

- **Headroom:** The number of available coordinators that trigger auto-scaling. For example, if Desired Free Capacity is set to 1 on an XSMALL-sized Virtual Warehouse, which has 2 executors, when there is less than one free coordinator (2 queries are concurrently executing), the warehouse auto-scales up and an additional executor group is added.
- **Wait Time:** How long queries wait in the queue to execute. A query is queued if it arrives on HiveServer and no coordinator is available. For example, if WaitTime Seconds is set to 10, queries are waiting to execute in the queue for 10 seconds. The warehouse auto-scales up and adds an additional executor group.

Auto-scaling based on wait time is not as predictable as auto-scaling based on headroom. The scaling based on wait time might react to non-scalable factors to spin up clusters. For example, query wait times might increase due to inefficient queries, not query volume.

When headroom or wait time thresholds are exceeded, the Virtual Warehouse adds executor groups until the maximum setting for Minimum and Maximum has been reached.

Procedure

1. In Concurrency Autoscaling properties, limit auto-scaling by setting the minimum and maximum number of executor nodes that can be added.



2. Choose when your cluster auto-scales up based on one of the following choices:

- Headroom
- Wait Time

3. Click Apply Changes.

Configuring Hive query isolation auto-scaling

You can configure your Hive Virtual Warehouse to add dedicated executors to run scan-heavy, data-intensive queries, also known as ETL queries. You learn how to enable a Virtual Warehouse auto-scaling feature and set a query isolation parameter in the Hive configuration.

Before you begin

- You are familiar with the [auto-scaling process](#).
- You are creating a Hive Virtual Warehouse for running ETL-type queries.
- In Cloudera Data Warehouse, you added a Hive Virtual Warehouse, configured the size of the Hive Virtual Warehouse, and configured auto-suspend as described in previous topics.
- You obtained the DWAdmin role.



Note: You can enable query isolation only while creating a Virtual Warehouse. On existing warehouses, you can only disable the query isolation option.

About this task

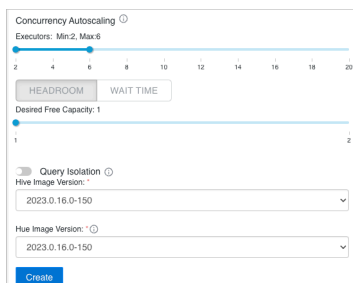
In this task, first you configure the same auto-scaling properties as described in the previous topic [Configuring Hive VW concurrency auto-scaling](#), and then you enable query isolation. Next, you set the `hive.query.isolation.scan.size.threshold` configuration parameter.

Procedure


1. In Concurrency Autoscaling properties, limit auto-scaling by setting the minimum and maximum number of executor nodes that can be added.



2. Choose when your cluster auto-scales up based on one of the following choices:
 - HEADROOM
 - WAIT TIME
3. In Concurrency Autoscaling, select Enable Query Isolation.



Two configuration options appear: Max Concurrent Isolated Queries and Max Nodes Per Isolated Query.

4. In Max Concurrent Isolated Queries, set the maximum number of isolated queries that can run concurrently in their own dedicated executor nodes.
Select this number based on the scan size of the data for your average scan-heavy, data-intensive query.
5. In Max Nodes Per Isolated Query, set how many executor nodes can be spawned for each isolated query.
6. Click Create Virtual Warehouse.
7. In the Cloudera Data Warehouse service **Overview** page, click Virtual Warehouses, and find your Virtual Warehouse. Click  Edit Configurations Hiveserver2 .
8. Select hive-site from the Configuration files drop-down menu and set hive.query.isolation.scan.size.threshold parameter to limit how much data can be scanned.
For example, set 400GB.
9. Click Apply Changes.

Configuring Impala Virtual Warehouse auto-scaling

Configuring the Impala Virtual Warehouse to use concurrency auto-scaling is critical for controlling cloud expenses.

Before you begin

- You are familiar with the [auto-scaling process](#).
- You are creating a Virtual Warehouse for running BI-type queries.
- In Cloudera Data Warehouse, you added an Impala Virtual Warehouse, configured the size of the Impala Virtual Warehouse, and configured auto-suspend as described in previous topics.
- You obtained the DWAdmin role.

About this task

In this task, you configure the following properties:

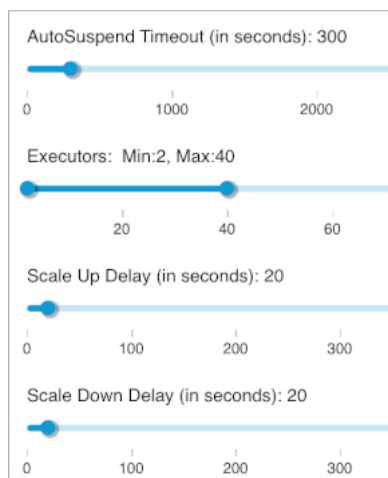
- Scale Up Delay: Sets the length of time in seconds to wait before adding more executors if queries wait in the queue.
- Scale Down Delay: Sets the length of time in seconds to wait before removing executors if executor groups are idle.

The time to auto-scale up or down is affected by the underlying Kubernetes configuration.

By default Impala Virtual Warehouses can run 3 large queries per executor group. Executors can handle more queries that are simpler and that do not utilize concurrency on the executor.

Procedure

1. In Scale Up Delay, set the seconds to wait before adding more executors if queries wait in the queue.



2. In Scale Down Delay, set the seconds to wait before removing executors when executor groups are idle.
3. Click Create Virtual Warehouse.

Configuring Workload Aware Autoscaling

You configure Workload Aware Autoscaling when you create a Virtual Warehouse following a step-by-step procedure using the Cloudera Data Warehouse UI.

About this task

You can also configure WAAS using the CLI.

Before you begin

- You are familiar with the [auto-scaling process](#) and [Workload Aware Autoscaling](#).
- You are creating a Virtual Warehouse for running BI-type queries.
- You obtained the DWAdmin role.

About this task

You perform the initial configuration of Workload Aware Autoscaling when you create a Virtual Warehouse, and later, you can edit your configuration. Tune your configuration based on meeting an SLA or reducing cost.

Procedure

1. Create an Impala Virtual Warehouse completing steps in [Creating a Virtual Warehouse](#) up to, and including, naming and selecting a Database Catalog for the Virtual Warehouse.
2. In Size, select use workload aware autoscaling.

3. In Small, which is an executor group set, configure Executors Per Group.

The number of executors per group is analogous to the number of nodes you would need on a bare metal system to handle your workload. A small executor group set helps improve cost and prevents larger groups from starting up.

4. In Large, which is your second executor group set, configure Executors Per Group.

Choose a value that is a multiple of the value you set for the small Executors Per Group. The greater the spread between values of Executors Per Group the better the query routing.

5. In Min No. of Groups, set the initial number of groups to use with no scaling down unless the Virtual Warehouse is suspended.
6. In Max No. of Groups, set the upper limit for scaling, which is often based on budget or intolerance for query latency.

The screenshot shows a configuration interface for executor groups. It has two main sections: 'Small' and 'Large'. Each section contains three input fields: 'Executors Per Group', 'Min. No. of Groups', and 'Max. No. of Groups'. Below each section is a link for 'Advanced Customizations' and a button for 'Add Custom Size'.

Group Set	Executors Per Group	Min. No. of Groups	Max. No. of Groups
Small	2	2	5
Large	20	0	1

An executor group set can contain up to 200 executors total. A group size (number of executors per group) must be within 1-100. The max no. of groups is limited by the group size. For example, a set with group size 2 can have up to 100 groups. A set with group size 100 can have up to 2 groups.

7. Optionally, click Advanced Customizations for an executor group to configure Disable AutoSuspend, AutoSuspend Timeout, Scale Up Delay, Scale Down Delay.
8. Optionally, click Add Custom Set to configure more scaling flexibility.
First, try using Workload Aware Autoscaling without a custom group. Later, if warranted, add up to three custom sets for a total of five executor group sets. Only use more than two executor groups under the following conditions:

- Your workload is well understood.
- You have a large mixed workload with clear categories of queries.

A row for configuring the Custom Set appears.

9. In Custom1, configure executors per group, minimum number of groups, and maximum number of groups.

Executors per group for a custom set cannot be less than executors per group for the small set or greater than executors per group for the large set.

The screenshot displays the configuration interface for custom executor group sets. It is divided into two main sections: 'Small' and 'Large'.

- Small Section:**
 - Executors Per Group: 2
 - Min. No. of Groups: 2
 - Max. No. of Groups: 5
 - Advanced Customizations: (collapsed)
- Large Section:**
 - Executors Per Group: 20
 - Min. No. of Groups: 0
 - Max. No. of Groups: 1
 - Advanced Customizations: (expanded)
 - ☐ Disable Auto-suspend ⓘ
 - AutoSuspend Timeout (in seconds): 300 ⓘ (slider from 0 to 7000)
 - Scale Up Delay (in seconds): 20 (slider from 0 to 1000)
 - Scale Down Delay (in seconds): 20 (slider from 0 to 1000)

Custom sized executor group sets are named group-set-custom-1, group-set-custom-2, and so on.

10. Run varying workloads, establish benchmarks for your queries, and add or delete custom sets.
11. Tune your configuration using the REQUEST_POOL query option or COMPUTE_PROCESSING_COST query option.

Query pool option example:

```
set request_pool=group-set-custom-1;
```

For more information, see [Request query pool option](#)

Restrict parallelism example:

```
Set COMPUTE_PROCESSING_COST=1;
```

For more information, see [Parallelism query option](#).

Configuring Impala coordinator shutdown

To cut cloud expenses, you need to know how to configure Impala coordinators to automatically shutdown during idle periods. You need to know how to prevent unnecessary restarts. Monitoring programs that periodically connect to Impala can cause unnecessary restarts.

About this task

When you create a Virtual Warehouse, you can configure Impala coordinators to automatically shutdown during idle periods. The coordinator start up can last several minutes, so clients connected to the Virtual Warehouse can time out.

Before you begin

- Update impyla, jdbc, impala shell clients if used to connect to Impala.
- When you create a Virtual Warehouse of SQL engine type Impala, and then you enable Unified Analytics, you can configure Impala coordinator shutdown only if you also configure Active-Passive mode.

High availability (HA) ⓘ

Enabled (Active-Passive) ▼

Executors ⓘ

2 200 Min 2 Max 40

Scale Up Delay (in seconds): 20

0 1000

Scale Down Delay (in seconds): 20

0 1000

Procedure

1. Follow instructions for "Creating a Virtual Warehouse".
2. Select a size for the Virtual Warehouse.
3. Do not select the Disable AutoSuspend option.

The Impala coordinator does not automatically shutdown unless the Impala executors are suspended.

4. Select the Allow Shutdown of Coordinator option.

After Impala executors have been suspended, the Impala coordinator waits for the time period specified by the Trigger Shutdown Delay before shutting down.

For example, if AutoSuspend Timeout = 300 seconds and Trigger Shutdown Delay=150 seconds, after 300 seconds of inactivity Impala executors suspend, and then 150 seconds later, the Impala coordinator shuts down.

5. Accept default values for other settings, or change the values to suit your use case, and click Create Virtual Warehouse.

Click the tooltip ⓘ for information about a setting.

Configuring Impala coordinator high availability

A single Impala coordinator might not handle the number of concurrent queries you want to run or provide the memory your queries require. You can configure multiple active coordinators to resolve or mitigate these problems. You can change the number of active coordinators later.

About this task

You can configure up to five active-active Impala coordinators to run in an Impala Virtual Warehouse. When you create an Impala Virtual Warehouse, Cloudera Data Warehouse provides you an option to configure Impala coordinator and Database Catalog high availability, described in the next topic. You can choose one of the following options:

Disabled

Disables Impala coordinator and Database Catalog high availability

Active-passive

Runs multiple coordinators (one active, one passive) and Database Catalogs (one active, one passive)

Active-active

Runs multiple coordinators (both active) and Database Catalogs (one active, one passive)

By using two coordinators in an active-passive mode, one coordinator is active at a time. If one coordinator goes down, the passive coordinator becomes active.

If you select the Impala coordinators to be in an active-active mode, the client software uses a cookie to keep a virtual connection to a particular coordinator. When a coordinator disappears for some reason, perhaps due to a coordinator shutting down, then the client software may print the error "Invalid session id" before it automatically reconnects to a new coordinator.

Using active-active coordinators, you can have up to five coordinators running concurrently in active-active mode with a cookie-based load-balancing.

An Impala Web UI is available for each coordinator which you can use for troubleshooting purposes.

Clients who connect to your Impala Virtual Warehouse using multiple coordinators must use the latest Impala shell. The following procedure covers these tasks.

Procedure

1. Follow instructions for "Creating Virtual Warehouse".
2. Select the number of executors you need from the Size dropdown menu.

A number of additional options are displayed, including High availability (HA).

High availability (HA) ⓘ

Enabled (Active-Passive) ▼

Executors ⓘ

2 200 Min 2 Max 40

Scale Up Delay (in seconds): 20

0 1000

Scale Down Delay (in seconds): 20

0 1000

3. Select the Enabled (Active-Active) option from the High availability (HA) drop-down menu.
4. Select the number of coordinators you need from the Number of Active Coordinators drop-down menu ranging from 2 to 5.


You can edit an existing Impala Virtual Warehouse to change the number of active coordinators.

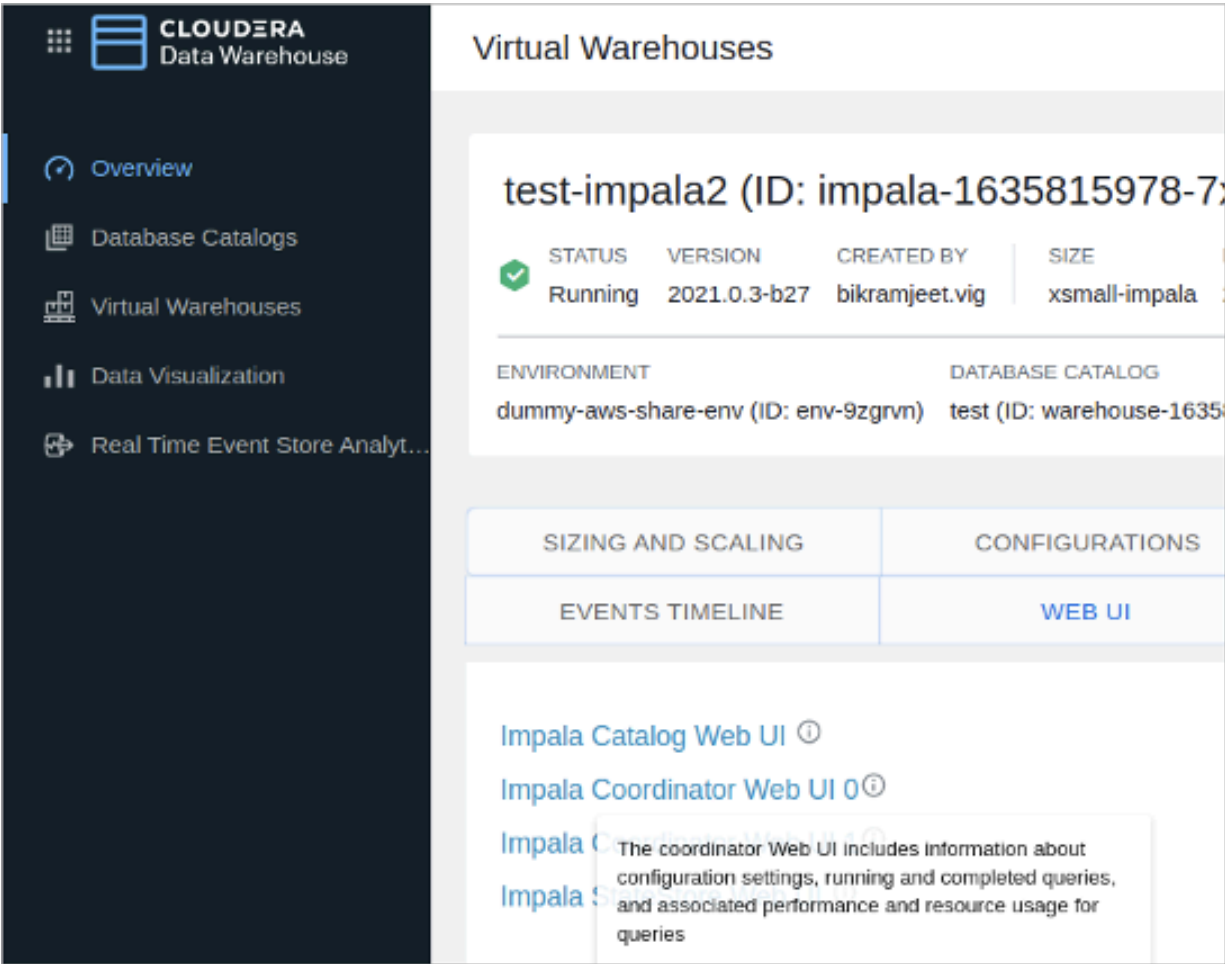



Important: Do not decrease the number of active-active coordinators you set up initially; otherwise, the Virtual Warehouse may shut down immediately. If clients are running queries on the Virtual Warehouse, the queries could fail.

5. Change values for other settings as needed, click Create Virtual Warehouse, and wait for the Impala Virtual Warehouse to be in the running state.

Click ⓘ to learn more about the setting.

6.
- Go to Cloudera Data Warehouse Overview Impala Virtual Warehouse  Edit Web UI , and then click each Impala Coordinator Web UI *N* link to get information about the coordinator.



7.
- Go to Overview Impala Virtual Warehouse  and select the Copy Impala shell Download command option. The following command is copied to your clipboard:

```
pip install impala-shell==4.1.0
```

8.
- Provide the command to clients who want to connect to the Impala Virtual Warehouse with multiple coordinators using the Impala shell.
9.
- Instruct the client user to update impyla to version compatible with Cloudera Data Warehouse, as listed in [Version information for Cloudera Data Warehouse on cloud components](#).
For example, installing/updating impyla 0.18a2, is required to connect to your Virtual Warehouse active-active coordinators in Cloudera Data Warehouse 2021.0.3-b27.
10.
- Inform the client that to connect over ODBC to an HA-configured Impala Virtual Warehouse that uses active-active coordinators, you must append impala.session.id to the HTTPAuthCookies connector configuration option of the Cloudera ODBC driver.

Table 7: HTTPAuthCookies

Key Name	Value	Required
HTTPAuthCookies	impala.auth,JSESSIONID,KNOXSESSIONID,impala.session.id	impala.session.id

Related Information

[Debugging Impala Virtual Warehouses using Web UIs](#)

Configuring Impala catalog high availability

By default, the Impala Virtual Warehouse runs a single instance of the catalog. The catalog stores databases, tables, resource usage information, configuration settings, and other objects managed by Impala. You can optionally configure running an additional Impala catalog instance. One catalog instance operates in active mode, the other in passive mode. The passive instance serves as a backup and takes over if the active instance goes down.

About this task

You enable catalog high availability when you create a New Virtual Warehouse. You cannot turn on, but you can turn off catalog high availability after creating an Impala Virtual Warehouse.

When you create an Impala Virtual Warehouse, you use the same UI dropdown to configure Impala coordinator, covered in the previous topic, and Database Catalog high availability. You can choose one of the following options:

- Disabled
Disables Impala coordinator and Database Catalog high availability
- Active-passive
Runs multiple coordinators (one active, one passive) and Database Catalogs (one active, one passive)
- Active-active coordinators
Runs multiple coordinators (both active) and Database Catalogs (one active, one passive)

Before you begin

You must obtain the DWAdmin role.

Procedure

1. Follow instructions for [Adding a new Virtual Warehouse](#).
2. In Size, select the number of executors, for example xsmall-2Executors.
A number of additional options appear, including High availability.

High availability (HA) ⓘ

Enabled (Active-Passive) ▼

Executors ⓘ

2 200 Min 2 Max 40


Scale Up Delay (in seconds): 20

0 1000

Scale Down Delay (in seconds): 20

0 1000

3. In High availability (HA), accept the default Enabled (Active-Passive) or Enabled (Active-Active).
Either option enables Database Catalog high availability in active-passive mode.
4. Accept default values for other settings, or change the values to suit your use case.

Click the tooltip  for information about settings.

5. Click Create Virtual Warehouse.

Configuring Impala Statestore high availability

You can configure High Availability (HA) for Impala Statestore pods in a Virtual Warehouse. One Statestore pod operates in active mode, the other in passive mode. The passive pod serves as a backup and takes over if the active pod goes down.

About this task



Note: You can enable Statestore HA only while creating a new Virtual Warehouse. Cloudera Data Warehouse provides you an option to turn off catalog and statestore HA on an existing Virtual Warehouse.

When you create an Impala Virtual Warehouse, you use the same UI dropdown to configure Impala coordinator, covered in the previous topic, and Database Catalog, and Statestore high availability. You can choose one of the following options:

Disabled

Disables Impala coordinator and Database Catalog, and Statestore high availability

Enabled (Active-passive)

Runs multiple coordinators (one active, one passive) and Database Catalogs, and Statestore (one active, one passive)

Enabled (Active-active)


Runs multiple coordinators (both active) and Catalog daemon (one active, one passive), and Statestore (one active, one passive)

Before you begin

You must obtain the DWAdmin role.

Procedure

1. Follow instructions for [Adding a new Virtual Warehouse](#).
2. In Size, select the number of executors, for example, xsmall - 2 Executors.
A number of additional options appear, including High availability.
3. In High availability (HA), accept the default Enabled (Active-Passive) or Enabled (Active-Active).
Either option enables Statestore high availability in active-passive mode.
4. Accept default values for other settings, or change the values to suit your use case.

Click the tooltip  for information about settings.

5. Click Create Virtual Warehouse.

Impala JDBC/ODBC host names

The host name prefix in a JDBC / ODBC URL of an Impala Virtual Warehouse depends on Impala coordinator configurations.

The following table shows the configuration combinations and resulting host name prefix in the JDBC/ODBC URL:

Coordinator HA	Allow Shutdown of Coordinator	Host Name Prefix
Disabled or enabled/active-passive	false	coordinator-<virtual-warehouse-name>
	true	impala-proxy-<virtual-warehouse-name>
Enabled/active-active	true or false	impala-proxy-<virtual-warehouse-name>

Creating a Virtual Warehouse with ARM compute instance types using CDP CLI

In Cloudera Data Warehouse 1.9.1-b233 (released July 26, 2024), you can select a compute instance type while creating, listing, or describing a Virtual Warehouse. By following the steps in this topic, you can create a Virtual Warehouse with regular or ARM-based instance types.

About this task

The following ARM-based instance types are supported starting with the 1.9.1-b233 release of Cloudera Data Warehouse on cloud:

Cloud provider	ARM instance type	Virtual Warehouse type
AWS	r7gd.4xlarge	Impala
	r6gd.4xlarge	
Azure	Standard_E16pds_v5	Impala

Before you begin

You must install the CDP CLI on your Kubernetes cluster.

Procedure

1. Connect to the Kubernetes cluster on which you want to create a Virtual Warehouse.
2. Run the following command to list the supported instance types:

```
dw describe-allowed-instance-types
```

3. Run the command to create a Virtual Warehouse:

For example:

```
dw create-vw --cluster-id <value> --dbc-id <value> --vw-type <value> --name <value> --instance-type <value>
```

Specify the instance type you want to use for your Virtual Warehouse using the `--instance-type` option.

For the complete list of options, see the documentation for the [create-dw command](#).

Configuring a Virtual Warehouse

After adding a Virtual Warehouse, you can change auto-scaling, enable single-sign-on (SSO) for your clients, configure a time zone, and speed up UDF queries by caching UDF JARs.

Tuning Virtual Warehouses

After creating a Virtual Warehouse, you can change the auto-scaling and some other configurations.

About this task

You set the auto-suspend timeout, the minimum and maximum number of executors for your virtual cluster, when your cluster should scale up and down, and other configurations.


Required role: DWAdmin

Procedure

1. Log in to the Cloudera web interface and navigate to the Cloudera Data Warehouse service.
2. In the Cloudera Data Warehouse service, click Overview in the left navigation pane.



Note: You can also tune your data warehouse on the Virtual Warehouse page, using the same steps.

3. In the **Overview** page under Virtual Warehouses, find your Virtual Warehouse, click , and select Edit.
4. In Sizing And Scaling, view the properties you can adjust to tune auto-scaling.
5. Configure auto-scaling as described in [Creating a Virtual Warehouse](#).
6. Click Apply Changes in the upper right corner of the page to apply your changes.

Related Tasks

[Tuning Impala auto-scaling](#)

Related Information

[Hive Virtual Warehouse auto-scaling](#)

[Hive query isolation for ETL jobs](#)

Tuning Impala auto-scaling

About this task

When you tune Impala Virtual Warehouses, you can disable the auto-suspend feature, set the minimum and maximum executor nodes allocated for the warehouse and you can set the scale up and scale down delay which determines auto-scaling behavior.

Required role: DWAdmin

Procedure

1. Log in to the Cloudera web interface and navigate to the Cloudera Data Warehouse service.
2. In **Overview**, in Virtual Warehouses, edit the Impala Virtual Warehouse.
3. Set the auto-suspend behavior:
 - If you do not want your Virtual Warehouse to auto-suspend, click Disable AutoSuspend. If you enable this feature, your Virtual Warehouse does not suspend itself, even if it is idle and no workloads are being processed.
 - If you do not want to disable auto-suspend, set the AutoSuspend Timeout. This sets the time, in seconds, that it takes for the Virtual Warehouse to automatically suspend itself. A Virtual Warehouse auto-suspends itself when the auto-scaler has scaled back to the last executor group and those executors are idle.

4. Adjust the minimum or maximum number of executor nodes as needed:

Setting and adjusting the minimum and maximum number of executor nodes per Virtual Warehouse is very similar to setting the number of nodes for on-premises clusters. Keep in mind the number of concurrent queries, the complexity of queries, and the volume of queries in your workloads to determine the appropriate number of executor nodes to set on each Virtual Warehouse instance.

5. Set the Scale Up Delay and the Scale Down Delay to fine-tune when the auto-scaler starts scaling up and the number of executor groups to meet workload demand.
 - Scale Up Delay: Sets the length of time in seconds that the system waits before adding more executors when it detects queries waiting in the queue to execute. The time to auto-scale up is affected by how the underlying Kubernetes system is configured.
 - Scale Down Delay: Sets the length of time in seconds that the system waits before it removes executors when it detects idle executor groups. As with the Scale Up Delay setting, the time to auto-scale down is affected by how the underlying Kubernetes system is configured.
6. Click Apply Changes.

Related Information

[Impala auto-scaling overview](#)

[Tuning Virtual Warehouses](#)

Tuning Impala Java tool options

After creating an Impala Virtual Warehouse, you can change the Java tool options. The options apply only to the Impala Catalog.

About this task


Tuning Impala Java tool options has the following limitations:

- The value of the JAVA_TOOL_OPTIONS represents JVM flags, but in this release, you can change only the Xmx and Xms values for CatalogD's JVM Heap.
- The Xmx flag value cannot be more than 18G.
- Floating point values for memory are not supported; only integers are supported.

Before you begin

Obtain the DWAdmin role.

Procedure

1. Log in to the Cloudera web interface and navigate to the Cloudera Data Warehouse service.
2. In the Cloudera Data Warehouse service, click Overview in the left navigation pane.
3. In the Overview page under Virtual Warehouses, select a Virtual Warehouse, click , and select Edit.
4. Click Configurations.
5. In Impala catalogd, in Configuration files, select env.
6. Adjust the value of environment variables JAVA_TOOL_OPTIONS for Impala CatalogD.
7. Click Apply Changes.

Disabling metadata synchronization

By default, metadata is synchronized when refreshed/invalidated across multiple Impala Virtual Warehouses that share a Database Catalog. When you run a command to refresh a table or invalidate metadata in any one of these Impala Virtual Warehouses, metadata is refreshed/invalidated in parallel. You learn how to disable this feature if you do not want to synchronize metadata for some reason.

About this task

By default, running a command from an Impala Virtual Warehouse to refresh tables or invalidate metadata raises events in the Hive metastore. Catalog daemons process events synchronously across all Virtual Warehouses that share metadata. Metadata is refreshed/invalidated in parallel across all your Virtual Warehouses. The `enable_reload_events` configuration parameter enables or disables raising events for metadata synchronization, effectively enabling or disabling the synchronization:

- `enable_reload_events = true` (default)
- `enable_reload_events = false`

Metadata has to be synchronized when you make changes to the data that causes inconsistent file metadata. For example, external clients create, update, and delete partitioned files on the object store.


To synchronize the file metadata in the catalog cache, you run a refresh/invalidate command in a single virtual warehouse. The catalog cache across all the Virtual Warehouses that share a Database Catalog is synchronized and consistent.

In this task, you disable metadata synchronization.

Before you begin

- The DWAdmin role is required to configure Cloudera Data Warehouse.
- An Impala Virtual Warehouse running 1.6.3-b319 (released May 5, 2023) or later


Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab and click  Edit .
The **Virtual Warehouse Details** page is displayed.
3. Go to Configurations Impala Catalogd .
4. Select flagfile from the Configuration files drop-down menu, and configure the `enable_reload_events` key to the value false.
5. Click Apply Changes.

Disabling Impala catalog high availability

You enable atalog high availability when you create a New Virtual Warehouse. You cannot turn on, but you can turn off catalog high availability after creating an Impala Virtual Warehouse. You turn off this feature by editing your Virtual Warehouse.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab and click  Edit .
The **Virtual Warehouse Details** page is displayed.
3. Go to the Sizing And Scaling tab and disable the Enable Impala Catalog Server HA option.
4. Click Apply Changes.

Enabling SSO to a Virtual Warehouse

Learn how to enable SSO (single sign-on) to your Virtual Warehouse from JDBC/ODBC clients. Your authorized clients can connect to the Virtual Warehouse using SSO. Find out how to recognize a connection string to connect to a Virtual Warehouse that is SSO-enabled.

About this task

You enable SSO connections to your Virtual Warehouse when you create a Virtual Warehouse. Authentication occurs through your browser and enterprise identity provider (IdP) provider. Your authorized clients can connect to the Virtual Warehouse using SSO.

When you configure an Impala Virtual Warehouse to use SSO, connections that use the JDBC URL or the ODBC connection string are SSO-enabled connections. Connections to the warehouse using the Impala shell will still use LDAP.


When you configure a Hive Virtual Warehouse to use SSO, all connections that use the JDBC URL are authenticated with the IdP provider that is configured in Cloudera Management Console.

If you are using ODBC to connect, ODBC connector (driver) documentation explains how to make the ODBC connection.

Before you begin

- You must configure an IdP in the [User Management module of Cloudera Management Console](#) compliant with Security Assertion Markup Language (SAML 2.0).
- In [Management Console User Management](#) you must set up a user group, required for enabling SSO, that identifies the users authorized to access to this Virtual Warehouse.
- You must obtain the DWAdmin role.

Procedure

1. Follow instructions for adding a [new Virtual Warehouse](#).
2. In **Size**, select the number of executors, for example `xsmall-2` Executors.
3. Under **Authentication**, select **Enable SSO** to enable single sign-on to your Virtual Warehouse, and in **User Groups**, select a user group set up in advance to access endpoints.
4. Accept default values for other settings, or change the values to suit your use case, and click **Create Virtual Warehouse**.
5. Optionally, find your Virtual Warehouses, click options , and select **Copy JDBC URL**.
The JDBC URL is copied to your clipboard.
6. Paste the URL somewhere, and check that a Virtual Warehouse is configured for SSO connections by looking for the “auth=browser” clause in the JDBC URL.
Hive example:

```
jdbc:hive2://hs2-test.env-9qls4k.dw.xcu2-8y8x.dev.cldr.work/default;transportMode=http;httpPath=cliservice;ssl=true;auth=browser;
```

Impala example:

```
jdbc:impala://coordinator-test.env-mxxmcn.dw.xcu2-8y8x.dev.cldr.work:443/default;AuthMech=12;transportMode=http;httpPath=cliservice;ssl=1;auth=browser;
```

7. [Give users the JDBC JAR and JDBC URL](#) information for connecting to the Virtual Warehouse.
Users configure an ODBC or JDBC connection to the Virtual Warehouse with a third-party BI tool. A browser window appears with the following success message:

```
Successfully authenticated. You may close this window.
```

Users can query tables they are authorized to access.

Configuring token-based authentication

Using a JSON web token (JWT), your Virtual Warehouse client user can sign on to your Virtual Warehouse for a period of time instead of entering single-sign on (SSO) credentials every time your user wants to run a query.

JWT authentication is a method of proving identity and is standardized through *IETF RFC 7519*. It uses a time limited token to provide identity instead of a username and password. JWT tokens are secrets that are generated by a third-party system using asymmetric key cryptography and used as a bearer token for accessing Virtual Warehouses. Virtual Warehouses only need the public portion of the third-party's asymmetric keys. Within Cloudera, Apache Knox serves as the trusted third-party JWT provider.

Following the procedure below, you acquire a token and set the lifespan of that token (cannot be revoked until expiration). Tokens are secrets that can be used multiple times until they expire.

If you created an Impala Virtual Warehouse for JWT authentication, your client user can choose the following ways to access your the Impala Virtual Warehouse:

- From Impyla

This option requires your client to configure Impyla as described in "Configuring Impyla for authentication".

- From a JDBC client.

If you set up a Hive Virtual Warehouse for authentication, you need to configure a few properties as described in "Configuring a Hive Virtual Warehouse for authentication". No configuration of the Impala Virtual Warehouse is required. Finally, you instruct your client to connect to your Hive or Impala Virtual Warehouse to use authentication from a JDBC client.


Related Information

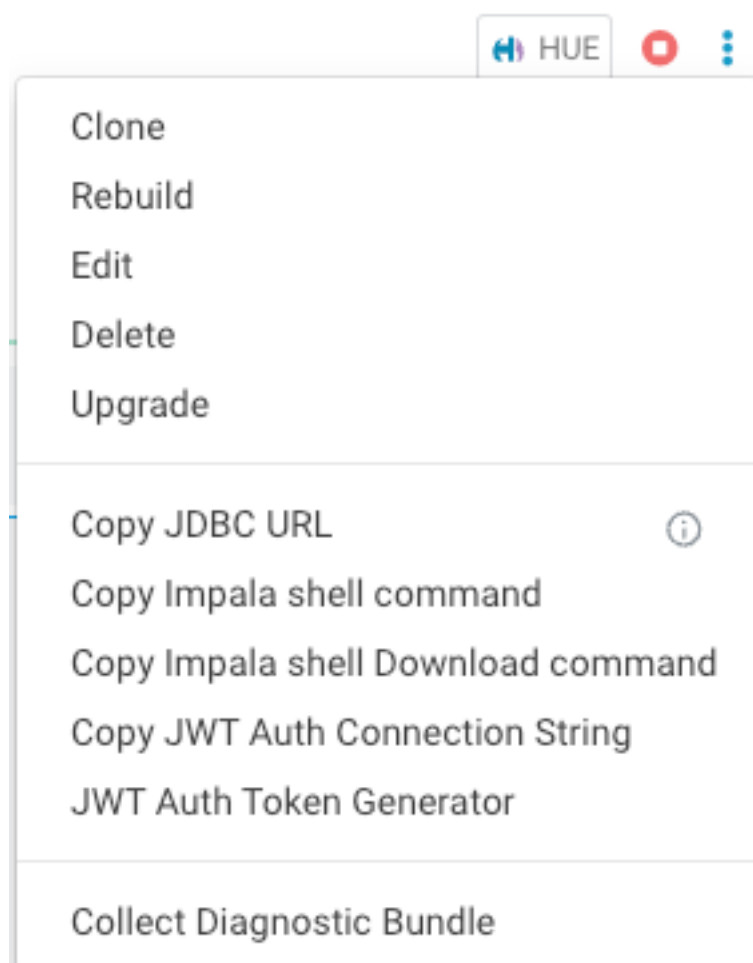
[IETF RFC 7519](#)

Generating a JWT token from the Cloudera side

You can generate a JWT token in Cloudera and use the token as a Standard Authorization Bearer token in your client to access a Hive or Impala Virtual Warehouse.

About this task

You can navigate to the Apache Knox UI in Cloudera, or click  in the Hive or Impala Virtual Warehouse tile. From the Virtual Warehouse tile, skip steps 1 and 2 below, and just click JWT Auth Token Generator shown in the following screenshot.

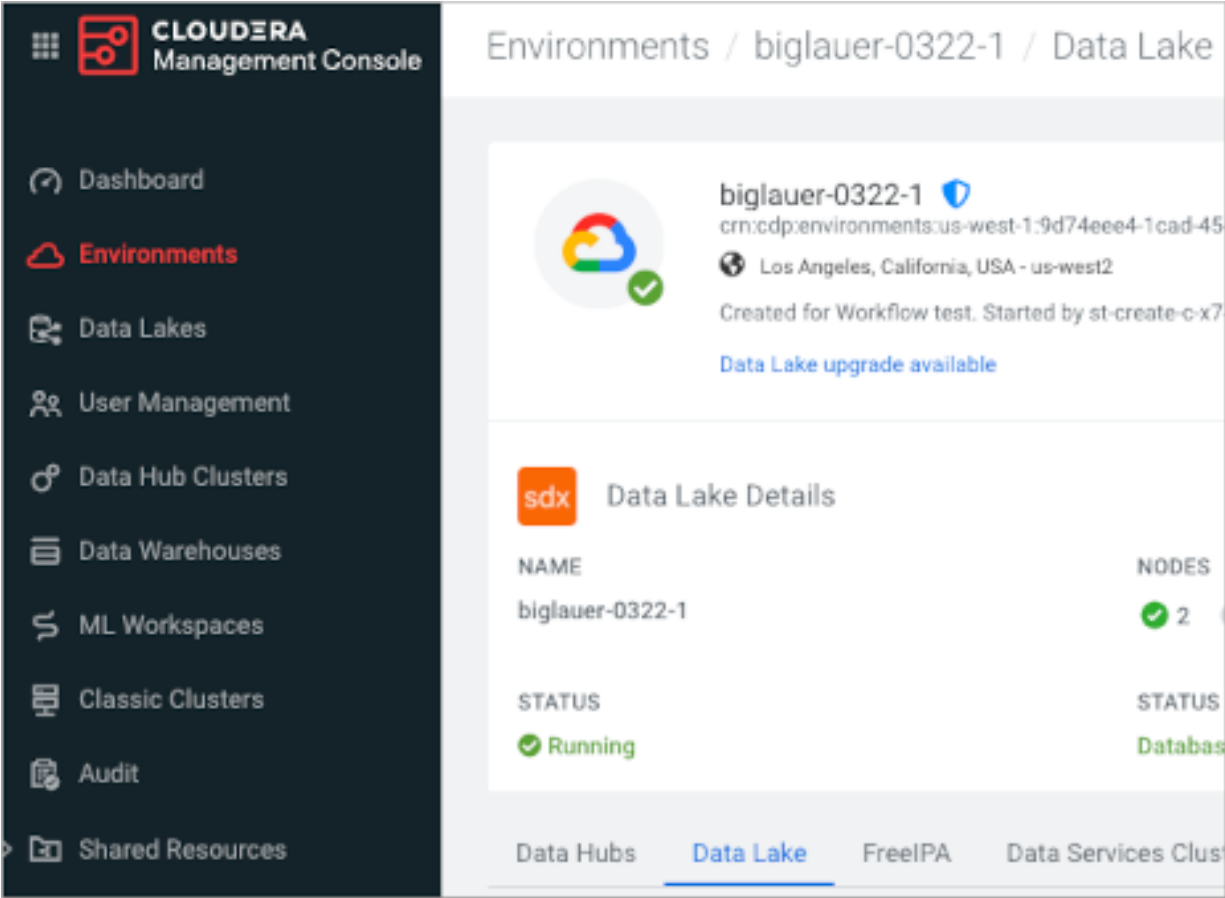


Apache Knox in Cloudera appears. Follow steps 3-5 below to generate the token.

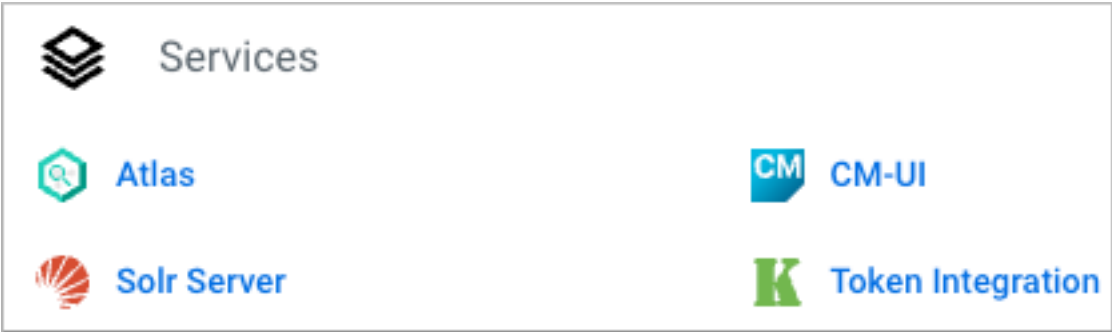
To navigate to the Knox UI instead of using the Cloudera Data Warehouse UI, follow the steps below:

Procedure

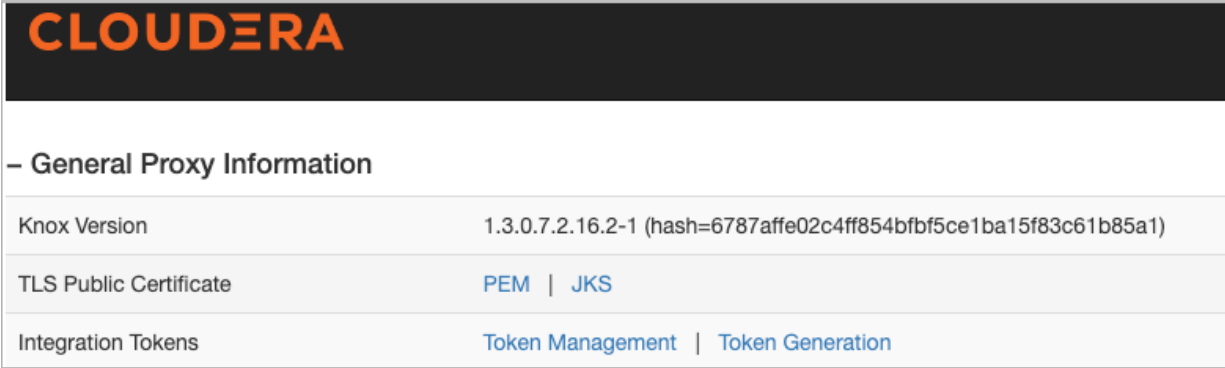
1. Log into Cloudera, and in the Cloudera Management Console, select Environments Data Lake .



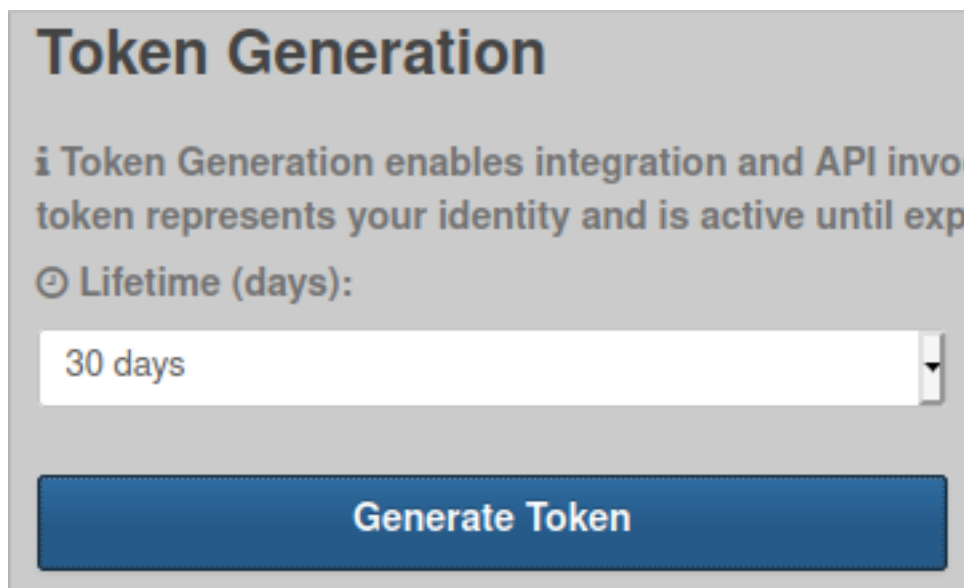
2. Scroll down to Services, and click Token Integration.



Apache Knox in Cloudera appears.



3. Click Token Generation.

A screenshot of a 'Token Generation' dialog box. At the top, the title 'Token Generation' is in large, bold, dark blue font. Below it, a paragraph explains: 'Token Generation enables integration and API invocation. A token represents your identity and is active until expiration.' Below this, there is a section labeled 'Lifetime (days):' with a clock icon. Underneath is a text input field containing '30 days'. At the bottom of the dialog is a large blue button with the text 'Generate Token' in white.

4. In **Token Generation**, choose the lifespan of the token, and click Generate Token.
5. Copy the JWT token that you generated and use the token on your Hive or Impala client to sign into your Virtual Warehouse.

For security reasons, after you close Token Generation, you cannot see the token again.

Configuring the Impala Virtual Warehouse for authentication

You need to enable platform JWT authentication for your Impala Virtual Warehouse during creation or editing of the Virtual Warehouse. This action automatically configures the Impala Virtual Warehouse to support authentication using JWT tokens issued by the Cloudera JWT authentication provider.

About this task

In this task, you see how to enable platform JWT authentication when you create a new Virtual Warehouse.

Before you begin

Do not enable Unified Analytics if you want to configure JWT access to your Impala Virtual Warehouse for clients.

Procedure

1. Follow the instructions for creating a Virtual Warehouse.
2. Under Authentication, select Enable JWT Authentication.

Configuring Impyla for authentication

You can connect Impyla to the Impala Virtual Warehouse using the HTTP transport protocol and JWT as the authentication mechanism.

Before you begin

Ensure you have Impyla configured for Impala. See, *Configuring Impyla for Impala*.

Procedure

1. In a terminal window, as a client, you run the command provided by the Impala Virtual Warehouse owner to update Impyla to the version compatible with the Impala Virtual Warehouse.

```
pip install impyla==0.18.0
```

2. Connect Impyla to the Impala Virtual Warehouse using a connection string as described below:

- host=Name of your Virtual Warehouse host
- port=Virtual Warehouse port
- use ssl=true

Cloudera Data Warehouse only provides a TLS endpoint, it does not provide an unencrypted endpoint. Also, not using TLS will expose the JWT in plain text.

- http_transport=True

The http transport is required by Impala for JWT authentication.

- jwt=<JWT token>

Substitute your JWT token, which you copied earlier, for <JWT Token>.

- http_path="cliservice"

Required.

```
self.connection = connect(<ENV>.host, <ENV>.http_port, use_http_transport=True,
http_path="cliservice",
auth_mechanism="JWT",
timeout=5, jwt=<JWT Token>)
```

On the client side, you can now run queries from Impyla on the Impala Virtual Warehouse.

Related Information

[Configuring Impyla for Impala](#)


Configuring the Hive Virtual Warehouse for authentication

You need to enable platform JWT authentication for your Hive Virtual Warehouse during creation or editing of the Virtual Warehouse and configure some HiveServer (HS2) properties. This ensures that the Hive Virtual Warehouse supports authentication using JWT tokens.

Before you begin

You must have acquired the JWT token through your Identity Provider (IDP). In Cloudera Data Warehouse, you use the Knox Token Generation service to generate the token.

Procedure


1. Follow the instructions for creating a Virtual Warehouse.
2. Under Authentication, select Enable JWT Authentication.
3. From the **Overview** page, go to the Virtual Warehouses tab, click  **Edit** against the required Hive Virtual Warehouse.
The **Virtual Warehouse Details** page is displayed.
4. Go to **Configurations Hiveserver2**.
5. Select hive-site from the Configuration files drop-down list, search for hive.server2.authentication, and set the value to JWT along with existing authentication protocols that are specified.
For example, 'LDAP,SAML,JWT'.
6. Search for hive.server2.authentication.jwt.jwks.skip.ssl.cert and set the value to true.
7. Search for hive.server2.transport.mode and set the value to http.

8. Search for `hive.server2.authentication.jwt.jwks.url` and, if you find it, set the value to the JSON Web Key Sets (JWKS) URL, which is used to sign the JWT token. For example, `https://myadmin-host/myadmin/homepage/knoxtoken/api/v1/jwks.json`.
If you do not find `hive.server2.authentication.jwt.jwks.url`, click Add Custom Configuration and add it to hive-site as a custom property, and then set the value to the JSON Web Key Sets (JWKS) URL.
You can fetch the JWKS URL by pasting the copied JWT token in the `jwt.io` website. Paste the JWT token in the Encoded text box of the Debugger and the decoded token header displays a `jku` parameter that contains the URL. You can provide a comma-separated list of `jwks.json` URL for this configuration.
9. If you modify the JWKS URL, restart HiveServer.
10. Click Apply Changes.

Getting a JDBC URL for your Virtual Warehouse

After generating a JWT token, if your clients access the Virtual Warehouse over a JDBC connection, you must get the URL to provide to the client.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab, locate the Virtual Warehouse you want to connect to, and click  Copy JDBC URL .
The JDBC URL is copied to your clipboard.
3. Paste the copied JDBC URL into a text file, and provide it to your client.
4. Instruct your client about how to use the token and JDBC URL as described in *Authenticating from a JDBC client*.

Authenticating from a JDBC client

On the Hive or Impala client side, use the JWT token as a Standard Authorization Bearer token by either specifying it in a JWT environment variable or by specifying the token directly in the JDBC connection string.

Before you begin

Authenticating from a JDBC client requires one of the following prerequisites:

- Connect from your client to a Hive or Impala Virtual Warehouse.
- Connect from your client to a Unified Analytics Virtual Warehouse of the Impala type using the Unified Analytics JDBC driver.

You can download the required driver from the Cloudera Data Warehouse Overview page. Click See More from the **Resources and Download** tile and select the required driver to download.



Note: Simba JDBC and ODBC drivers for Impala support JWT authentication in this release. Ensure that you download the required Database drivers from [Cloudera Downloads](#).

Procedure

Specify the JWT token you copied in "Generating a JWT token from the Cloudera side" as a JWT environment variable, and then construct a Beeline command.

Hive example:

```
export JWT=[***JWT token***]

beeline -u "jdbc:hive2://<your-virtual-warehouse>.<your-environment>.<dwx.
company.com>/default;transportMode=http;httpPath=cliservice;ssl=true;retries
=3;auth=jwt"
```

Impala example using the Unified Analytics JDBC driver and a Unified Analytics Impala Virtual Warehouse:

```
export JWT=[***JWT token***]

beeline -u "jdbc:impala://<your-virtual-warehouse>.<your-environment>.<dwx.com>.<company.com>/default;transportMode=http;httpPath=cliservice;ssl=true;retries=3;auth=jwt"
```

Alternatively, specify the JWT token directly in the JDBC URL connection string.

Hive example:

```
beeline -u "jdbc:hive2://<your-virtual-warehouse>.<your-environment>.<dwx.com>.<company.com>/default;transportMode=http;httpPath=cliservice;ssl=true;retries=3;jwt=[***JWT token***]"
```

Impala example using the Unified Analytics JDBC driver and a Unified Analytics Impala Virtual Warehouse:


```
beeline -u "jdbc:impala://<your-virtual-warehouse>.<your-environment>.<dwx.com>.<company.com>/default;transportMode=http;httpPath=cliservice;ssl=true;retries=3;jwt=[***JWT token***]"
```

You can now query the Virtual Warehouse from the client side.

Connecting Cloudera Data Warehouse and Cloudera AI

A Cloudera user can connect to Cloudera Data Warehouse from Cloudera AI using a JWT Auth Connection String.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab, locate the Virtual Warehouse you want to connect to, and click  Copy JWT Auth Connection String .
The JWT connection string is copied to your clipboard.
3. Paste the copied string into a text file, and provide it to your Cloudera AI client user.
For more information about connecting Cloudera AI to Cloudera Data Warehouse Virtual Warehouses, see the Cloudera AI documentation.

Configuring a time zone

Cloudera Data Warehouse by default is configured for the UTC time zone. If you want time-related queries in your Virtual Warehouse to return results in your local time zone instead of UTC, you need learn how to change the configuration.

About this task


The way you configure the time zone in the Hive Virtual Warehouse versus the Impala Virtual Warehouse differs. In this task, you configure both.

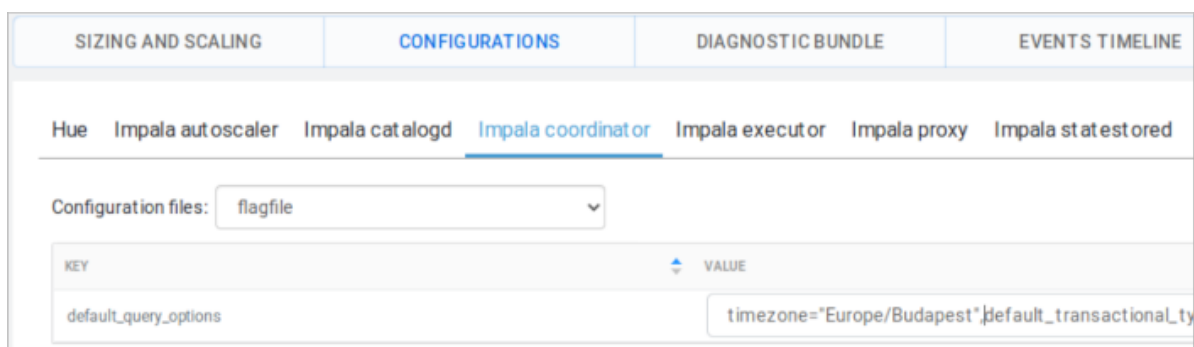
- First you configure a Hive Virtual Warehouse to return Dubai time when you enter a time-related query, such as `SELECT CURRENT_TIMESTAMP`.
- Next, you configure an Impala Virtual Warehouse to return Budapest time when you enter a query such as `SELECT NOW()`.



Note: Configuring a timezone in a Virtual Warehouse affects only the time-related queries and not the logs. The Cloudera Data Warehouse logs use the UTC timezone.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab and click  Edit against your Virtual Warehouse.
The **Virtual Warehouse Details** page is displayed.
 - a) (Hive) Go to Configurations HiveServer2 .
 - b) Select hive-site from the Configuration files drop-down list, and search for hive.local.time.zone.
 - c) If the hive.local.time.zone key does not appear, click Add Custom Configuration to add it.
 - d) In Configuration Value, enter your local time zone.
Refer to [Timezonedb](#) to find the name of your time zone. For example, in Configuration Value, specify Asia/Dubai.
 - a) (Impala) Go to Configurations Impala Coordinator .
 - b) Select flagfile from the Configuration files drop-down list, and to the default_query_options key, add your local time zone.
For example, in VALUE, add timezone="Europe/Budapest",.



SIZING AND SCALING		CONFIGURATIONS	DIAGNOSTIC BUNDLE	EVENTS TIMELINE
Hue Impala autoscaler Impala catalogd Impala coordinator Impala executor Impala proxy Impala statelord				
Configuration files: flagfile				
KEY		VALUE		
default_query_options		timezone="Europe/Budapest",default_transactional_ty		

3. Click Apply Changes.

Configuring UDF JAR caching in Hive Virtual Warehouse

After you write and compile your User Defined Function (UDF) code into a Java Archive (JAR) file, you can configure a Hive Virtual Warehouse to cache the UDF JAR in HiveServer (HS2) in Cloudera Data Warehouse.

About this task

UDFs enable you to create custom functions to process records or groups of records. Although Hive provides a comprehensive library of functions, there are gaps for which UDFs are a good solution.

In this task, you configure the Hive Virtual Warehouse to cache the JAR file for quick access by the Virtual Warehouse. After configuring the Virtual Warehouse for caching, the UDF JAR is downloaded from the object store the first time it is called, and then cached. Subsequent calls to the JAR are answered from the cache.


Configuring caching significantly improves performance for queries that use the UDF. Without caching, loading a very large UDF of several hundred MBs can take up to several minutes for each query.

Before you begin

- Create a user-defined function.
- Write, compile, and export your UDF code to a JAR file.
- Upload the UDF JAR file to a bucket or container on AWS or Azure, respectively.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.

2. Go to the Virtual Warehouses tab, locate the Hive Virtual Warehouse that uses the bucket or container where you placed the UDF JAR file, and click  Edit .
The **Virtual Warehouse Details** page is displayed.
3. Go to Configurations HiveServer2 .
4. Select hive-site from the Configuration files drop-down menu, and click Add Custom Configuration.
The **Custom Configuration** modal is displayed.
5. Add the following configuration information, and then click Add:
 - Configuration Key: hive.server2.udf.cache.enabled
 - Configuration Value: true
6. Click Apply Changes.
7. Verify that the configuration property and setting have been added by searching for hive.server2.udf.cache.enabled in the search box. If the property has been added, the property name is displayed in the KEY column of the table.

Related Information

[Creating a user-defined function](#)

Connecting Cloudera Data Warehouse and Kudu

You can configure an Impala Virtual Warehouse to connect to Kudu in Data Hub and then create tables stored in Kudu from Impala. You learn prerequisites, gather information for the configuration, and see the steps for making the connection. There are several advantages of running Kudu queries from Cloudera Data Warehouse.

Running Kudu queries from an Impala Virtual Warehouse provides benefits, such as [isolation](#) from noisy neighbors, [auto-scaling](#), and [autosuspend](#). After configuring an Impala Virtual Warehouse to connect to Kudu, you can create Kudu tables using Hue or the Impala shell.

Prerequisites

You must meet the following requirements:

- Obtain the DWAdmin role.
- Get permission to access Cloudera Data Hub 7.2.15, or later, details to obtain Kudu cluster information.
Cloudera Data Hub cluster 7.2.15, or later, is required
- Reactivate your AWS or Azure environment version to get 1.6.1-b258 with runtime 2023.0.13.20 (released Feb 7, 2023) and use an Impala Virtual Warehouse.



Note: You can upgrade an existing Impala Virtual Warehouse with an older runtime version to 2023.0.13.20 or later. Set catalogd flagfile configuration enable_kudu_impala_hms_check to false if you upgrade.

Gathering Kudu master FQDNs

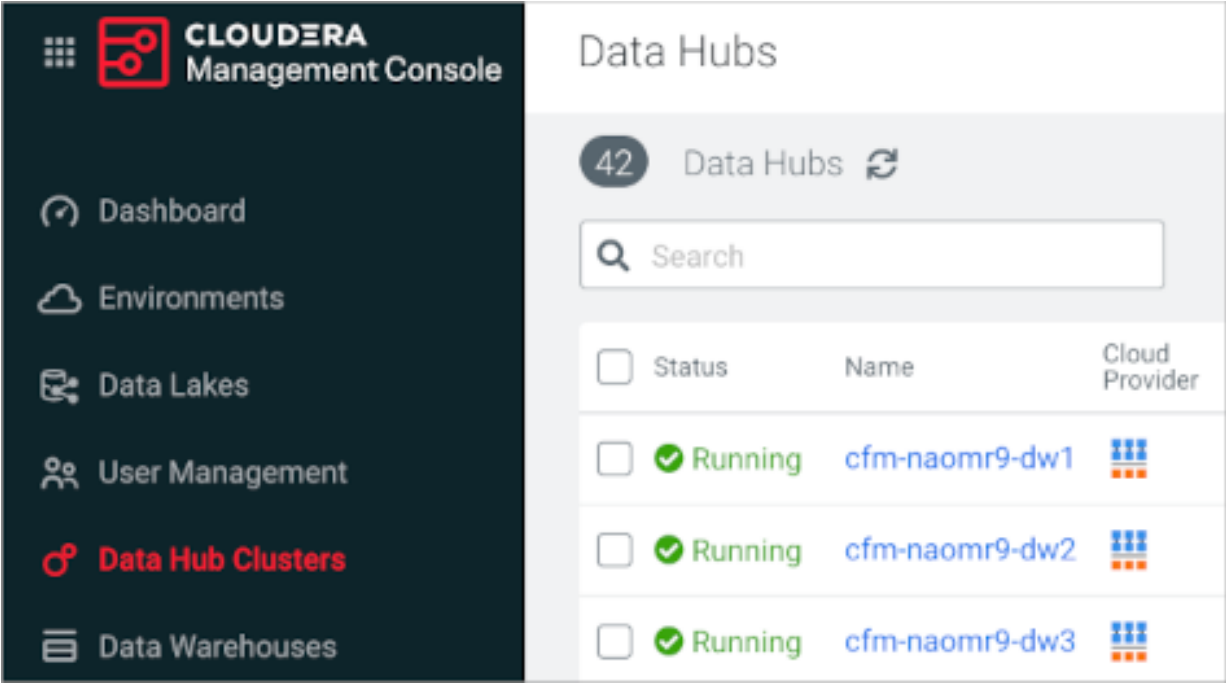
You need to get the fully qualified domain names (FQDNs) of the Kudu master hosts in your Cloudera Data Hub cluster. You follow steps to navigate to the Kudu Master service and find the FQDNs for configuring a Cloudera Data Warehouse-Kudu connection.

Before you begin

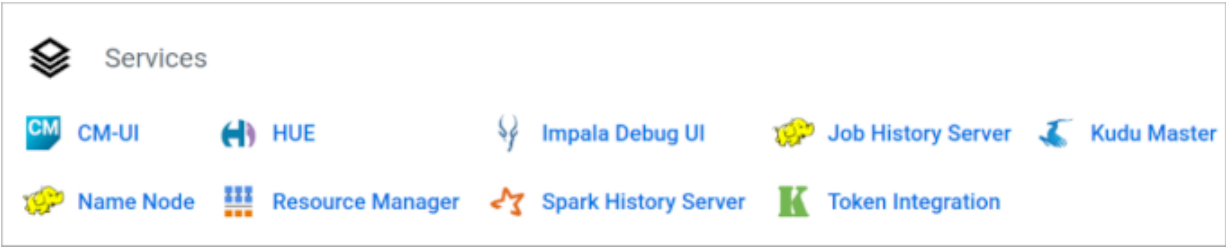
You must meet the prerequisites listed in the previous topic, "Connecting Cloudera Data Warehouse and Kudu".

Procedure

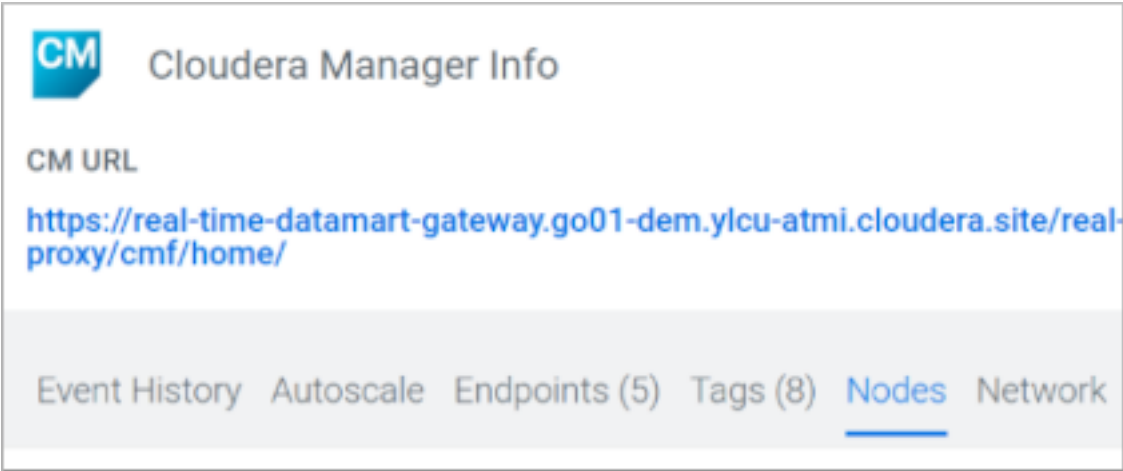
- 1. Login to Cloudera and click Cloudera Data Hub Clusters.



- 2. In Cloudera Data Hub, in Name, click a cluster name.
- 3. Scroll down to Services.






- 4. Click Kudu Master, and note the instance ID of a Kudu master host(s).
- 5. Scroll down to Cloudera Manager Info, and click Nodes.



The list of nodes appear, including their fully qualified domain name (FQDN).

- Find the Kudu cluster in the list, and note the FQDN of the instances(s) running Kudu Master services.

For example:

Event History Autoscale Endpoints (5) Tags (8) <u>Nodes</u> Network Load Balance					
Master2					
<input type="checkbox"/>	Instance ID		Status	FQDN	
<input checked="" type="checkbox"/>	 i-044bdd1bb0be50eec		 Running	pm-kudu-master	

Now, you have the information you need to configure a Cloudera Data Warehouse-Kudu connection as described in the next topic.


Configuring a Cloudera Data Warehouse-Kudu connection

The steps for making the connection involve setting a Impala coordinator property to the fully qualified domain names (FQDNs) of the Kudu master hosts.

Before you begin

- You must meet the prerequisites listed in the "Connecting Cloudera Data Warehouse and Kudu".
- You need to have gathered the FQDNs of Kudu master hosts as described in the previous topic, "Gathering Kudu master FQDNs".

Procedure

- From the Cloudera Management Console or Cloudera landing page, navigate to Data Warehouse.
- In Virtual Warehouses, select your Impala Virtual Warehouse, and click  Edit .
- In **Virtual Warehouse Details** page, click Configurations Impala Coordinator .

4. Select flagfile from the Configuration files drop-down list, and set the kudu_master_hosts key to a list of all Kudu master FQDNs.

New Virtual Warehouse example:

```
kudu_master_hosts=kudu-master1.sandbox.a.cloudera.site:7051,kudu-master2.sandbox.a.cloudera.site:7051,kudu-master3.sandbox.a.cloudera.site:7051
```

The screenshot shows a web interface titled "Virtual Warehouses". Below the title is a table with two columns: "KEY" and "VALUE". The first row has the key "kudu_master_hosts" and the value "host1:7051, host2:7051, host3:7051". The second row has the key "invalidate_tables_timeo" and the value "3600".

KEY	VALUE
kudu_master_hosts	host1:7051, host2:7051, host3:7051
invalidate_tables_timeo	3600

This screenshot shows the three FQDNs, oversimplified for readability.

Existing Virtual Warehouse:

- Click Add Custom Configurations, set kudu_master_hosts to the FQDN of a single master host as shown follows:
 - Configuration Key: kudu_master_hosts
 - Configuration Value: host1:7051
 - Select flagfile from the Configuration files drop-down list, and add the FQDNs of other Kudu master hosts to the value of kudu_master_hosts.
5. Click Apply Changes.

When the Virtual Warehouse finishes updating, you can query Kudu tables from Hue, an Impala shell, or an ODBC/JDBC client. For example:

```
--- First, create an example table
CREATE TABLE my_first_kudu_table
(
  id BIGINT,
  name STRING,
  PRIMARY KEY(id)
)
PARTITION BY HASH PARTITIONS 16
STORED AS KUDU;

--- Next, insert some example data
INSERT INTO my_first_kudu_table VALUES (101,'A'),(102,'B'),(103,'C');
--- Finally, select from the example table to show the above has worked
SELECT * FROM my_first_kudu_table;
```

Setting up a past release for a Cloudera Data Warehouse-Kudu connection

To connect Cloudera Data Warehouse to Kudu using a past release 1.5.1-b110 (released Nov 22, 2022) or earlier, you need to configure Kubernetes DNS resolution unless you create a new environment. Configuring the Kubernetes DNS resolution is required only in AWS environments.

About this task

If you create a new environment, you do not need to configure Kubernetes DNS resolution. After completing the steps in this task, you need to meet additional prerequisites and configure a Cloudera Data Warehouse-Kudu connection as described in the previous topic, “Configuring a Cloudera Data Warehouse-Kudu connection”.

Procedure

1. If you have an AWS environment, perform all the steps in this procedure. If you have an Azure environment, go to step 6 (upgrading).
2. [Grant remote access to Kubernetes clusters on Amazon EKS.](#)
3. On the command line of your cluster, open the Cloudera CoreDNS Updater file for configuration in a text editor. For example:

```
kubectl edit configmap cdp-coredns-updater-63cc3d36-283cd77b-template -n cluster
```

The template will have a unique name, not `cdp-coredns-updater-63cc3d36-283cd77b-template` in your cluster. The data section looks something like this:

```
data:
Corefile.template: |-
.:53 {
    errors
    health
    kubernetes cluster.local in-addr.arpa ip6.arpa {
        pods insecure
        fallthrough in-addr.arpa ip6.arpa
        ttl 30
    }
    prometheus :9153
    forward . /etc/resolv.conf
    cache 30
    loop
    reload
    loadbalance
}
{{applications.FREEIPA.services.dns.config.domain}}:53 {
    errors
    cache 30
    reload
    forward . {{#applications.FREEIPA.services.dns.endpoints}}{{host}} {/{
applications.FREEIPA.services.dns.endpoints}}
```

4. Append a section to the data section above to enable reverse lookup for data lake IPs.

```
cluster.local in-addr.arpa ip6.arpa {
    kubernetes {
        pods insecure
        fallthrough in-addr.arpa ip6.arpa
        ttl 30
    }
    forward . {{#applications.FREEIPA.services.dns.endpoints}}{{host}} {/{
applications.FREEIPA.services.dns.endpoints}}
```

Your data section should now look like this:

```
data:
Corefile.template: |-
.:53 {
    errors
    health
    kubernetes cluster.local in-addr.arpa ip6.arpa {
    pods insecure
    fallthrough in-addr.arpa ip6.arpa
    ttl 30
    }
    prometheus :9153
    forward . /etc/resolv.conf
    cache 30
    loop
    reload
    loadbalance
    }
    {{applications.FREEIPA.services.dns.config.domain}}:53 {
        errors
        cache 30
        reload
        forward . {{#applications.FREEIPA.services.dns.endpoints}}{{host}} {{/
applications.FREEIPA.services.dns.endpoints}}
    }
    cluster.local in-addr.arpa ip6.arpa {
        kubernetes {
        pods insecure
        fallthrough in-addr.arpa ip6.arpa
        ttl 30
        }
    }
    forward . {{#applications.FREEIPA.services.dns.endpoints}}{{host}} {{/app
lications.FREEIPA.services.dns.endpoints}}
    }
}
```

- Restart the coredns updater pod to pick up the new configurations.

```
kubectl --kubeconfig ~/kubeconfig_int get pods -n cluster | grep -i 'cor
edns-updater'
cdp-coredns-updater-63cc3d36-283cd77b-67b8c9479d-bjtlh 3/3 Running 1 (2d20
h ago) 2d20h

kubectl --kubeconfig ~/kubeconfig_int delete pod cdp-coredns-updater-63cc3
d36-283cd77b-67b8c9479d-bjtlh -n cluster
```

- If the Impala Virtual Warehouse runtime version is 2022.0.12.0-90 released December 13, 2022 or earlier, upgrade the Virtual Warehouse version to 2023.0.13.20 or later.
- Proceed to configure a Cloudera Data Warehouse-Kudu connection as described in the previous topic, "Configuring a Cloudera Data Warehouse-Kudu connection".

Configuring Iceberg manifest caching in Impala Virtual Warehouse

Apache Iceberg provides a mechanism to cache the contents of Iceberg manifest files in memory. In Cloudera Data Warehouse, you can enable or disable Iceberg manifest caching for Impala Coordinators and Catalogd, and set a few other properties, in your Impala Virtual Warehouse.

About this task

The [manifest caching feature](#) helps to reduce repeated reads of small Iceberg manifest files from remote storage by Impala Coordinators and Catalogd.

The following default properties are set in Cloudera Data Warehouse for manifest caching:

```
iceberg.io.manifest.cache-enabled=true;
iceberg.io.manifest.cache.max-total-bytes=104857600;
iceberg.io.manifest.cache.expiration-interval-ms=3600000;
iceberg.io.manifest.cache.max-content-length=8388608;
```

The following list describes each property:

- `iceberg.io.manifest.cache-enabled`: enable/disable the manifest caching feature.
- `iceberg.io.manifest.cache.max-total-bytes`: maximum total amount of bytes to cache in the manifest cache. Must be a positive value.
- `iceberg.io.manifest.cache.expiration-interval-ms`: maximum duration for which an entry stays in the manifest cache. Must be a non-negative value. Setting zero means cache entries expire only if it gets evicted due to memory pressure from `iceberg.io.manifest.cache.max-total-bytes`.
- `iceberg.io.manifest.cache.max-content-length`: maximum length of a manifest file to be considered for caching in bytes. Manifest files with a length exceeding this property value will not be cached. Must be set with a positive value and lower than `iceberg.io.manifest.cache.max-total-bytes`.


Generally, you set a different value for the expiration interval in catalogd and the coordinator. The expiration time is later in catalogd, for example, 1 week. Catalogd needs caching for a longer period of time because the catalogd service serves table metadata.

Changing configuration parameters in Cloudera Data Warehouse is recommended only when following Cloudera instructions.

Before you begin

Obtain the DWAdmin role.

Procedure

1. In Cloudera Data Warehouse, click Overview, and find your Impala Virtual Warehouse.
2. Click  Edit Configurations Impala Catalogd
3. Search for the following string: `iceberg.io`.

The following properties appear:

SIZING AND SCALING

CONFIGURATIONS

DIAGNOSTIC BUNDLE

EVENTS

Hue

Impala autoscaler

Impala catalogd

Impala coordinator

Impala executor

Impala proxy

Impala token auth

Configuration files:

hadoop-core-site-default-warehouse

KEY	VALUE
iceberg.io.manifest.cache-enabled	true
iceberg.io.manifest.cache.expiration-interval-ms	604800000
iceberg.io.manifest.cache.max-content-length	8388608
iceberg.io.manifest.cache.max-total-bytes	104857600

✓ Apply Changes

Discard Changes

4. Change the value of one or more properties, and click Apply Changes.

- Click **Configurations Impala coordinator**.

The following properties appear:

SIZING AND SCALING	CONFIGURATIONS	DIAGNOSTIC BUNDLE	EVENTS TIM
Hue	Impala autoscaler	Impala catalogd	Impala coordinator
	Impala executor	Impala proxy	
Impala token auth			
Configuration files: hadoop-core-site-default-warehouse			
KEY	VALUE		
iceberg.io.manifest.cache-enabled	true		
iceberg.io.manifest.cache.expiration-interval-ms	3600000		
iceberg.io.manifest.cache.max-content-length	8388608		
iceberg.io.manifest.cache.max-total-bytes	104857600		

- Change the value of one or more properties, and click **Apply Changes**.

Managing high partition workloads

You learn how to identify an error related to high partition workloads that require configuration of your Hive Virtual Warehouse to run successfully. You might need to configure your Virtual Warehouse to prevent these errors when inserting data into workloads with a large number of partitions and columns.


About this task

To prevent an error when inserting data into a high partition workload, such as a table having 5,000 partitions and 100 columns, configure the Hive Virtual Warehouse as described in the steps below.


The error message you might see when inserting data into a high partition workload that is not configured properly looks something like this:

```
ERROR : FAILED: Execution Error, return code 40000 from org.apache.hadoop.hive.ql.exec.MoveTask. MetaException(message:One or more instances could not be made persistent)
```

Procedure

- Try [changing the size](#) of your Hive Virtual Warehouse.
For example, if you experience the high partition error in a small Virtual Warehouse, try using a medium Virtual Warehouse and set min/max nodes to 40.
- In the Cloudera Data Warehouse UI, in Overview, go to the Virtual Warehouses tab, click , and select **Edit** corresponding to your Virtual Warehouse.
- Click **Configurations HiveServer2**.
- Tune HS2 parameters in the Virtual Warehouse.
For example, search for and set the following properties, or if the property is not found, click **Add Custom Configuration** to add a custom configuration, and set it as follows:

```
set hive.optimize.sort.dynamic.partition.threshold=0;
set hive.thrift.client.max.message.size=2147483647;
set hive.metastore.client.skip.columns.for.partitions=true;
set hive.stats.autogather=false;
set hive.stats.column.autogather=false;
set hive.msck.repair.batch.size=[***TABLE SCHEMA SIZE***];
```

- Go to the **Database Catalogs** tab, and select your Database Catalog.
- Click **Options** , and select **Edit**.

7. Tune Metastore parameters in the Database Catalog.

For example, click **Configurations Metastore**, search for and set the following properties, or if the property is not found, click + to add a custom configuration, and set it as follows.

```
hive.metastore.direct.sql.batch.size=[***TABLE SCHEMA SIZE***]
hive.txn.timeout=3600
hive.metastore.try.direct.sql=true
hive.metastore.try.direct.sql.ddl=true
```

8. Click **Apply Changes**.

Configuring an Operational Database connection

Having a connection to Cloudera Operational Database, you can create tables from Hive and store data in HBase.

The tables you create from Hive and store in HBase have some metadata stored in the Hive metastore. The data is stored in HBase.

You must meet the following requirements to configure a Cloudera Operational Database connection.

- Have an AWS-based environment.
- Have administrative access to the S3 object store associated with your Hive Virtual Warehouse.

Configuring a Cloudera Operational Database connection involves three top-level steps. First, you set up Ranger permissions to access Cloudera Operational Database from Hive. Next, you generate configuration files. Finally, you configure the Hive Virtual Warehouse to connect to Cloudera Operational Database.


Setting up Cloudera Data Warehouse to connect to Cloudera Operational Database

You configure your Virtual Warehouse to use configuration files you generated for connecting to Cloudera Operational Database.

Before you begin

- You generated configuration files `hbase-site.xml` and `hbase-cdw.tar.gz` as described in the previous topic.

Procedure

1. In the Cloudera Data Warehouse UI, in **Overview**, go to the **Virtual Warehouses** tab, select a Hive Virtual Warehouse, and click  **Edit**. The **Virtual Warehouse Details** page is displayed.
2. Go to **Configurations Hiveserver2**.
3. In **Configuration files**, select `hbase-site.xml`, and copy the contents of the `hbase-site.xml` generated by the script you ran.
4. Create a directory or use an existing directory on S3.

```
hadoop fs -mkdir s3a://odx-weekly-test/cod--tddl6gutqgn9/jars
```

5. Copy the `hbase-cdw.tar.gz`, which you also obtained from the script you ran, to that directory.
6. In the **Hive Virtual Warehouse Details Configurations Hiveserver2**, in **Configuration files**, select `env` and select the `CDW_HIVE_AUX_JARS_PATH` key.
7. Set the value of the `CDW_HIVE_AUX_JARS_PATH` key to the path of the jar file.
8. Go to the **Query coordinator** tab and select `env` from the **Configuration files** drop-down menu, and select the `CDW_HIVE_AUX_JARS_PATH` key.
9. Set the value of the `CDW_HIVE_AUX_JARS_PATH` key to the path to the jar file.
10. Click **Save Changes** and restart the Virtual Warehouse.

Resizing the Impala Virtual Warehouse size

After creating an Impala Virtual Warehouse, you can tune, or correct, the T-shirt size of executor groups that drive the Impala Virtual Warehouse. The size of the executor groups is critical for achieving cost and performance goals.

About this task

You do not have to recreate the Virtual Warehouse if you want to try a different size. For example, you can easily change the size of executor groups from xxsmall-impala to xsmall-impala after creating a the Virtual Warehouse by following the procedure below to edit the Impala Virtual Warehouse.

Before you begin


- You obtained the DWAdmin role.
- Understand "[Warehouse sizing requirements](#)".

Procedure

1. Log in to the Cloudera web interface and navigate to the Cloudera Data Warehouse service.
2. In the Cloudera Data Warehouse service, click Overview in the left navigation pane.



Note: You can also tune your data warehouse on the Virtual Warehouse page, using the same steps.

3. In **Overview** page, find your Virtual Warehouses, click , and select Edit.
4. In Sizing And Scaling, select a different size from the Size drop-down menu.
5. Click Apply Changes.

Resizing the Hive Virtual Warehouse size

The size of the Hive Virtual Warehouse you select during Virtual Warehouse creation determines the number of executors and concurrent queries the Virtual Warehouse can run. You need to know how to change the size of the Virtual Warehouse upward or downward to tune performance and manage cost.

About this task

You cannot change the size of a Hive Virtual Warehouse, but you can handle incorrect sizing in the following ways.

- You can delete the Virtual Warehouse, and then recreate it in a different size.
- You can change the auto-scaling thresholds to change the effective size of the Virtual Warehouse based on demand. The actual size does not change, but increases or decreases in resources occurs automatically.


This task assumes you have two Virtual Warehouses that you decide are incorrectly sized for some reason. You correct the sizing of one by deleting and recreating the Virtual Warehouse. You correct the effective sizing of the other by changing auto-scaling thresholds.

Before you begin

- You obtained the DWAdmin role.

Procedure

First Virtual Warehouse: Replace this Virtual Warehouse

1. Log in to the Cloudera web interface, navigate to Data Warehouse Overview , note the name of the Virtual Warehouse you want to modify, and note which Database Catalog it is configured to access.
2. Click the options  of the Virtual Warehouse you want to delete, and select Delete.
3. Go to the Virtual Warehouses tab and click New Virtual Warehouse.
4. Set up the new Virtual Warehouse:


- Type the same Name for the new Virtual Warehouse as you used for the old Virtual Warehouse.



Note: The fully qualified domain name of your Virtual Warehouse, which includes the Virtual Warehouse name plus the environment name must not exceed 64 characters; otherwise, Hue cannot load.

- In Type, click the SQL engine you prefer: Hive or Impala.
 - Select your Database Catalog and User Group if you have been assigned a user group.
 - In Size, select the number of executors, for example xsmall-2Executors.
 - Accept default values for other settings, or change the values to suit your use case.
5. Click Create Virtual Warehouse.

Second Virtual Warehouse: Change the Auto-Scaling Thresholds

6. In Data Warehouse Overview , click the options  of the other Virtual Warehouse, a Hive Virtual Warehouse for example, to change auto-scaling thresholds, and select Edit.
7. Go to the Sizing And Scaling tab and in Concurrency Autoscaling, slide the control to change the Max number of executors.
8. Click Apply Changes.

Related Information

[Creating a Virtual Warehouse](#)

[Configuring auto-scaling](#)

Compaction in Cloudera Data Warehouse

You understand the importance of compaction and the consequences of neglecting to perform compaction. Compaction keeps your Data Warehouse healthy.

Over time tables belonging to a workload become fragmented due to operations performed on them by your workload users. These small, obsolete files might lead to performance degradation and query latency problems. Compaction plays a major role in improving response time to workload queries by reducing the number of underlying files for a table and eliminating the obsolete ones. Compaction runs periodically in the background to maintain the optimal state.

Running periodic compaction is a best practice for the performance for ACID transactions. ACID inserts and deletes generate the problematic files that you might need to monitor and manage. In Cloudera Data Warehouse, compaction is always performed by a Hive Virtual Warehouse.

Compaction prerequisites

To prevent data loss or an unsuccessful compaction, you must check for adequate resources in your first Hive Virtual Warehouse used for compaction.

Check for adequate resources

You must take into account the query workload for compaction when you create the first Hive Virtual Warehouse. You must make sure that the warehouse has adequate resources to handle the compaction workload in addition to any other workloads you might run in that warehouse.



Important: Impala Virtual Warehouses cannot be designated as the compactor Virtual Warehouse for a Database Catalog. Compaction tasks can only be assigned to a Hive Virtual Warehouse.

Change compactor configuration for Hive Virtual Warehouses on Cloudera Data Warehouse on cloud

To enhance performance, the compactor is a set of background processes that compact delta files, which are created as a by-product of data modifications. When it runs, it incurs additional load on the Hive Virtual Warehouse assigned as the compactor in Cloudera Data Warehouse on cloud. You can change which Hive warehouse performs compaction to load-balance this workload as necessary.


About this task

Required role: DWAdmin

Before you begin

One Hive Virtual Warehouse must be the compactor for all the Virtual Warehouses associated with a particular Database Catalog. This Hive Virtual Warehouse compactor runs all of the compaction queries for all Virtual Warehouses that use one particular Database Catalog, including Impala Virtual Warehouses. However, Impala Virtual Warehouses cannot be designated as the compactor Virtual Warehouse for a Database Catalog. Compaction tasks can only be assigned to a Hive Virtual Warehouse. The first Hive Virtual Warehouse you create against a Database Catalog is automatically set as the compactor. If you decide you do not want that particular warehouse to take on the compaction workload, you can set another Hive Virtual Warehouse to perform the compaction workload by following these steps:

Procedure

1. Log in to the Cloudera web interface and navigate to the Cloudera Data Warehouse service.
2. On the Overview page, select the Hive Virtual Warehouse that you want to set as the compactor, and click the options menu .
3. In the options menu, select Set Compactor.


Related Information

[Virtual Warehouse sizing requirements for public cloud environments](#)

Initiating automatic compaction

To start automatic compaction in Cloudera Data Warehouse, you set a property in the Database Catalog.

Procedure

1. Log into the Cloudera web interface and in Cloudera Home, click Cloudera Data Warehouse.
2. In Cloudera Data Warehouse, select your Database Catalog and click  Edit.
3. Click Configurations Metastore.
4. From the drop-down, select hive-site for the configuration category.
5. Set `hive.compactor.initiator.on` to true.
6. Click Apply Changes and restart the Database Catalog.


Configuring compaction

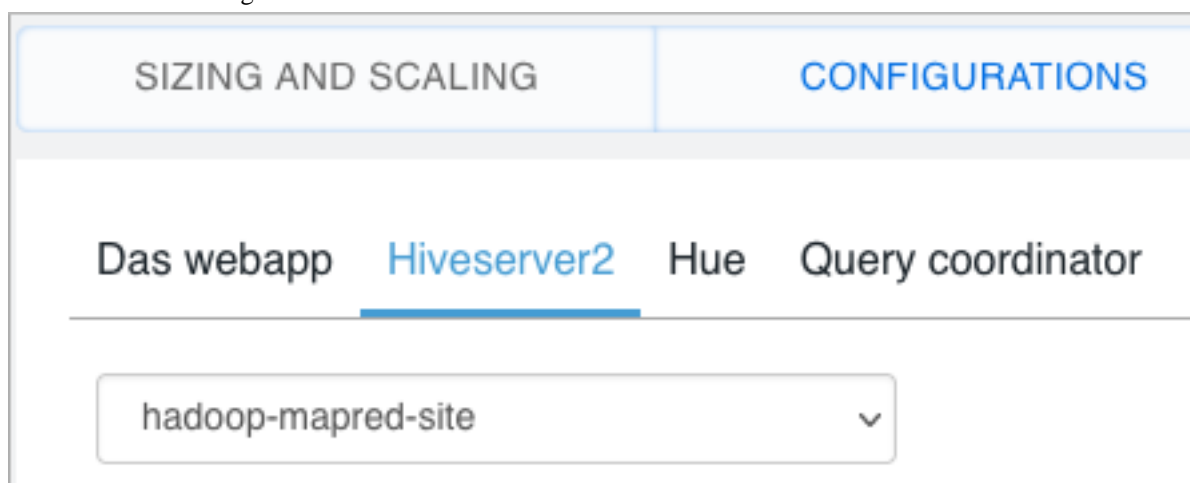
You configure some configure compaction properties from the Database Catalog and some from the Virtual Warehouse.

About this task

In this task, you configure the compactor worker threads in the Virtual Warehouse.

Procedure

1. Go to the Virtual Warehouses tab and select a Hive Virtual Warehouse.
2. Click  Edit Configurations Hiveserver2 .



3. Select hive-site from the Configuration files drop-down menu.
4. View, increase, or decrease the number the hive.compactor.work.threads.
5. Click Apply Changes.

Automating metadata invalidation after compaction

As you insert or delete data from Hive ACID tables, Hive generates delta files. The compaction process consolidates the delta files and keeps the system healthy. If you query Hive ACID tables from an Impala Virtual Warehouse, you need to learn how to automatically invalidate metadata after compaction to prevent a possible query failure.


About this task

After compaction of ACID tables, metadata in Impala coordinator local caches might be stale. You might experience a failure (FileNotFoundException) when you query these tables. Invalidation refreshes the metadata and prevents the problem. You can configure automatic metadata invalidation as described in the following procedure.

Before you begin

You must upgrade your Database Catalog to the compatible version for DWX-1.1.2-b2008 or later to use this feature.

Procedure

1. In Cloudera Data Warehouse, click Overview.
2. In an Impala Virtual Warehouse, click  Edit .
3. Go to Configurations Impala Coordinator and select flagfile from the Configuration files drop-down menu.

4. Set `auto_check_compaction` to true, and click Apply Changes.

How compaction works

When data changes are made on Cloudera Data Warehouse with inserts, updates, and deletes, delta files are created. The more changes that are made, the more delta files are created. When a large number of delta files are created, query performance degrades. Compaction removes these delta files to enhance query performance.

There are two types of compaction:

Minor compaction

Compacts multiple delta files into a single delta file.

Major compaction

Compacts one or more delta files and the base file for the bucket and creates a single new base file per bucket.

The goal of compaction is to "self heal" tables in order to restore the baseline query performance. All compactions are done in the background and do not prevent concurrent reads and writes of the data. After compacting, the system waits for all readers of the old files to finish and then removes the old files.

Compactor processes

These background processes run inside the metastore and HiveServer2 in Cloudera Data Warehouse on cloud. They support the data modifications made as a result of ACID transactions.

Compactor process	Description
Initiator	<p>This process runs in the metastore, which equates to the Database Catalog construct in the Cloudera Data Warehouse UI, and discovers which tables and partitions are due for compaction. By default, it runs every 5 minutes. To change this interval:</p> <ol style="list-style-type: none"> 1. Identify the Database Catalog for the Virtual Warehouse on which you want to change the compaction interval by selecting the Virtual Warehouse tile. The associated Database Catalog is highlighted. 2. In the Database Catalog, click the edit icon in the tile to launch the Database Catalog details page. 3. On the Database Catalog details page, make sure the CONFIGURATIONS tab is selected, and then select the Metastore subtab. 4. On the Metastore tab, select <code>hive-site</code> from the drop-down list, and search for the <code>hive.compactor.check.interval</code> KEY. 5. Add your preferred check interval in the associated VALUE field in seconds. 6. Click Apply Changes. The services are automatically updated with the new configuration.
Worker	<p>This process runs in HiveServer2, which equates to the Hive Virtual Warehouse construct in the Cloudera Data Warehouse UI. The worker process performs the actual compacting work. In Cloudera Data Warehouse, compaction runs an INSERT statement created from the output of a SELECT statement, thereby re-writing the data to new base or delta files.</p>
Cleaner	<p>This process runs in the metastore and deletes delta files after compaction and after it determines the files are no longer needed. By default, the cleaner runs every 5 seconds (5,000 milliseconds). The check occurs on the visibility ID/transaction ID, which is a global transaction identifier.</p>

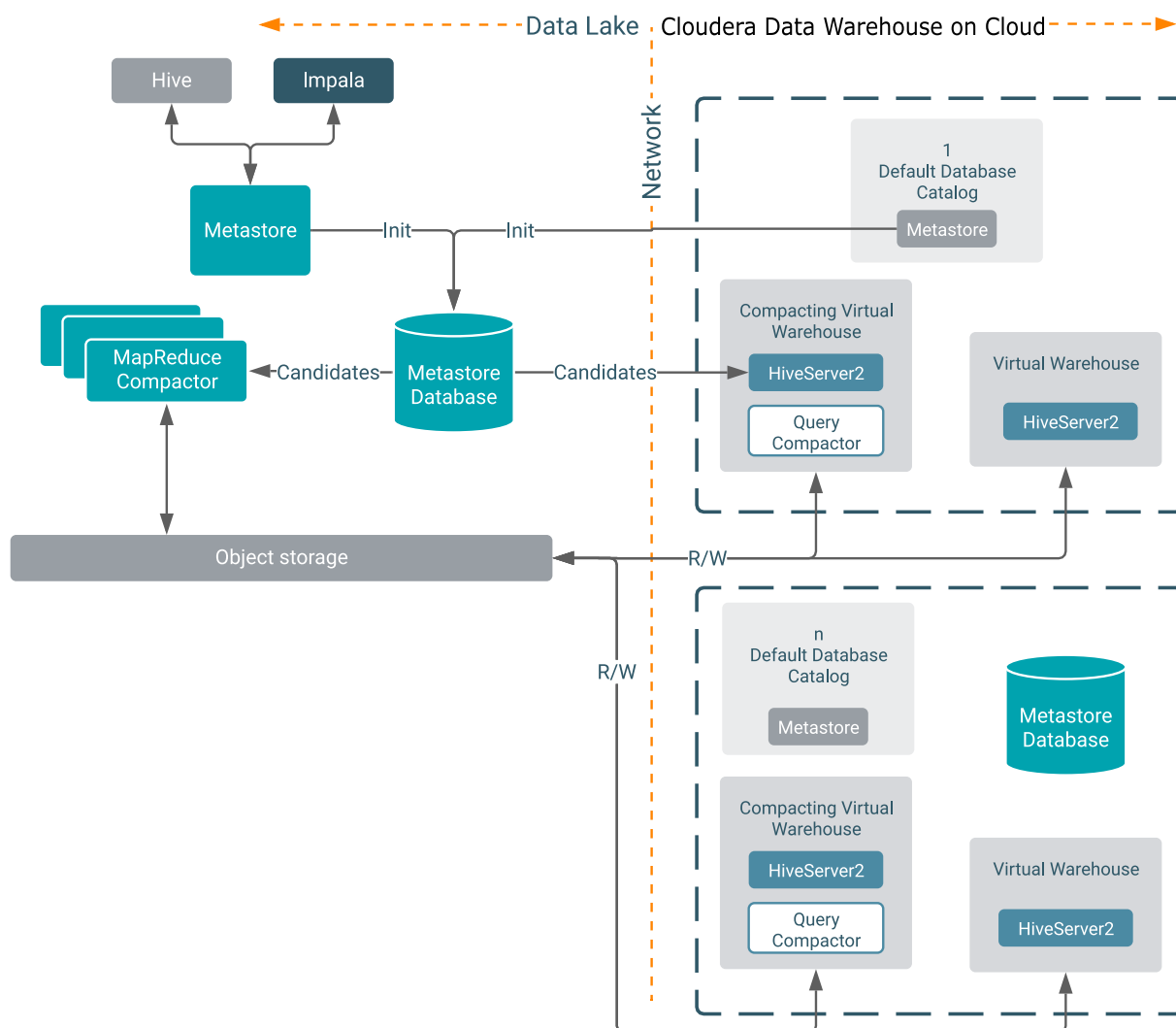
How compaction interacts with the Data Lake

In the Data Lake on Cloudera, the initiator and cleaner processes also run in the metastore as they do in Cloudera Data Warehouse on cloud. However, the worker process runs in HiveServer (HS2), which equates to a Hive Virtual Warehouse..

In Cloudera Data Warehouse, the initiator and cleaner processes run in the Database Catalog, which equates to the metastore. The Database Catalog maintains a connection with the Data Lake and compaction jobs run in parallel with it. The worker process in Hive Virtual Warehouses executes queries to perform compaction.

Cloudera Data Warehouse on cloud Compaction Architecture

This diagram illustrates how the components that perform compaction interact on Cloudera Data Warehouse on cloud.



Compaction observability

Compaction observability is a notification and information system based on metrics about the health of the compaction process. A healthy compaction process is critical to query performance, availability, and uptime of your data warehouse. You learn how to use compaction observability to prevent serious problems from developing.

Compaction runs in the background. At regular intervals, Hive accesses the health of the compaction process and logs an error in the event of a problem. The assessment is based on metrics, such as the number of rows in the metadata table `TXN_TO_WRITE_ID` and the age of the longest running transaction (`oldest_open_txn_age_in_sec`). For example, if compaction isn't running the `TXN_TO_WRITE_ID` table in the HMS backend database becomes bloated and queries slow down. You use prebuilt Grafana dashboards to view alerts about compaction status, the issue, and recommended actions. The following list describes a few of more than 25 notifications:

- Oldest initiated compaction passed threshold
- Large number of compaction failures

- More than one host is initiating compaction

Compaction alerts use metrics to provide the following information to help you proactively address the problems before the problems become an emergency:

- Warnings and errors that suggest next steps
- Charted metrics
- Hive logging

Compaction observability does not attempt to do root cause analysis (RCA) and does not attempt to fix the underlying problem. Compaction observability helps you quickly respond to symptoms of compaction problems. Factors unrelated to compaction per se can look like a compaction problem. For example, an underlying problem related to renewing a Kerberos ticket problem can surface as a compaction problem. Configuring kerberos to add authorization, changing the running user, or increasing the queue size might solve the problem. Compaction observability provides troubleshooting information.

Compaction alerts are enabled by default in Cloudera Data Warehouse and the compaction health data is collected by default. Alerts place no load on Hive. The data about compaction health is not stored for very long, and is not stored in Hive. The data is emitted from Hive, and a backend thread, which is configurable to run as often as you want, collects metrics in Prometheus.

Create an environment in this release to pick up this feature.

Viewing a compaction alert using Grafana

Learn how to access Grafana dashboards from Cloudera Data Warehouse to view compaction alerts and take necessary actions to keep your cluster healthy.



Before you begin

You must be able to log into Grafana from Cloudera Data Warehouse as described in the *Getting started in Grafana* topic.

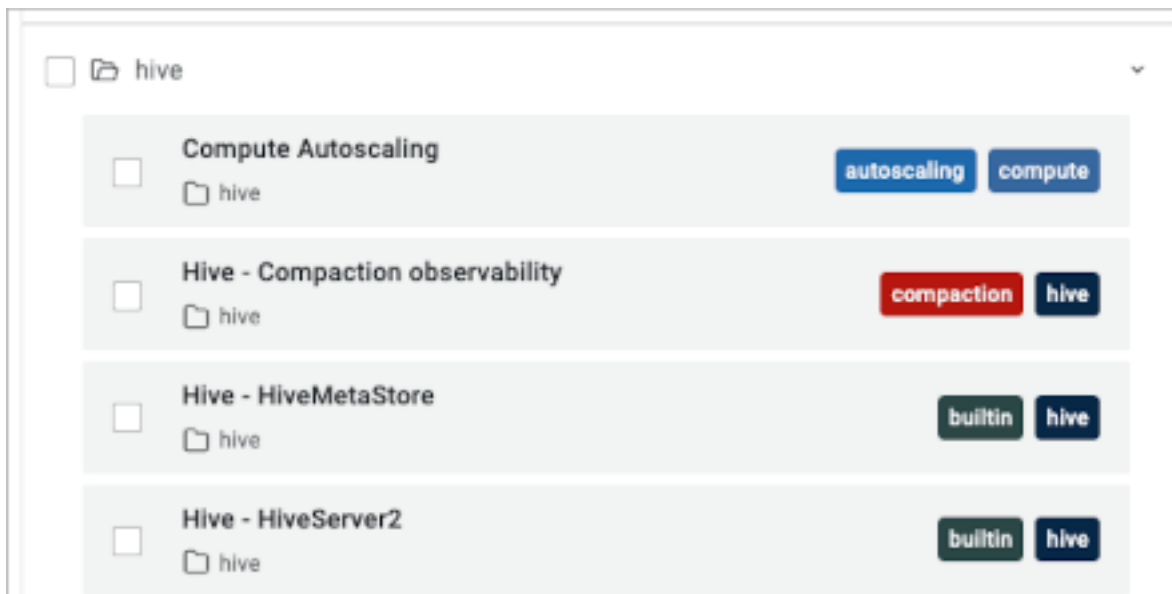
About this task

Perform the following steps to access Grafana from Cloudera Data Warehouse:

Procedure

1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate the environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.
3. Click  Open Grafana .
4. In the Grafana web interface, click  and then select Dashboards.
A list of dashboard groups is displayed.

5. Locate and click the Hive - Compaction observability dashboard group to view all the dashboards related to Hive compactions.



Related Information

[Getting started in Grafana](#)

[Viewing prebuilt Grafana dashboards](#)

Adding a compaction alert

You can define compaction alert rules for your Cloudera Data Warehouse deployment based on Prometheus expressions. The alerts are automatically triggered when specific events occur in your deployment.

About this task

You use PromQL expressions to express the conditions that trigger the alert, as shown in the following procedure.

Procedure

1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate an environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.
3. Click Edit .
4. In the **Environment Details** page, click the ALERT SETTINGS tab and then click Add New.
5. Enter the following information for the new alert:
 - Alert ID
 - Name of the alert
 - An error or warning message to display when the alert is triggered
 - The Prometheus expression to specify conditions that trigger the alert
 - Duration to specify how long the conditions last before the alert is triggered
 - Severity of the alert — Error or Warning
6. Click Create.



Related Information




[Querying Prometheus](#)

Managing a compaction alert

You can manage compaction alerts in Cloudera Data Warehouse by modifying alert details, such as the alert name, message, expression that defines threshold conditions for the alert, frequency, and severity of the alert.

Procedure

1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate an environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.
3. Click  Edit .
4. In the **Environment Details** page, click the ALERT SETTINGS tab and then click  Edit for the alert that you want to modify.

Severity	Name	Summary
 Warning	LargeNumberOfCompactionFailuresWarning	Large number of compaction failures
 Warning	OldestInitiatedCompactionPassedThresholdWarning	Oldest initiated compaction passed threshold
 Error	OldestInitiatedCompactionPassedThresholdError	Oldest initiated compaction passed threshold

Edit

Delete

5. In the **Edit Alert Details** page, modify the required properties to redefine the alert.

Edit Alert Details

X

ALERT ID

LargeNumberOfCompactionFailuresWarning

Name

Large number of compaction failures

Message

A high percentage of compactions have failed or were not queued. To find error messages, run `SELECT * FROM SYS.COMPACTIONS WHERE C_STATE IN ("failed", "did not initiate");` Check Virtual Warehouse or Data Hub logs for "Caught exception while trying to compact"

Expression

$$\frac{(\max(\text{compaction_num_failed}) \text{ by } (\text{namespace}, \text{service}) + \max(\text{compaction_num_did_not_initiate}) \text{ by } (\text{namespace}, \text{service}))}{(\max(\text{compaction_num_failed}) \text{ by } (\text{namespace}, \text{service}) + \max(\text{compaction_num_did_not_initiate}) \text{ by } (\text{namespace}, \text{service}) + \max(\text{compaction_num_succeeded}) \text{ by } (\text{namespace}, \text{service}))} > 0.1$$

Duration

0s ⓘ

Severity

Warning ▼

Update

Cancel

6. Click Update.


Configuring alert notifications

As Administrator, you need to learn how to configure alert notifications that appear in the user notification system. The notifications supplement alert information in charts on the Manage/Hive/Compaction Observability dashboard in Grafana. You learn to use the UI for this configuration, but alternatively, you can use the Kubernetes configuration method.

About this task

In this task, you see how to funnel alerts into a user notification system. In the next task, you see how to use the Kubernetes configuration method.

Procedure

1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate an environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.
3. Click  Edit .
4. In the **Environment Details** page, click the ALERT SETTINGS tab.
5. In Alert Manager Configuration, configure the receiver object by following instructions in the [Prometheus AlertManager v.0.20.0 documentation](#) to configure the receiver object.

For example, a simple WebHook configuration looks something like this:

```
global:
  resolve_timeout: 5m

route:
  group_by: ['alertname', 'namespace', 'service']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 3h
  receiver: notification_system

receivers:
- name: 'notification_system'
  webhook_configs:
  - url: https://co-alerting.prod.mycompany.com
    send_resolved: true

inhibit_rules:
- source_match:
  severity: 'error'
  target_match:
  severity: 'warning'
  equal: ['alertid', 'namespace', 'service']
```

For information about integration possibilities, see the [receiver section](#).

6. Click Save, and then Apply Changes.
The alertmanager recognizes the change and automatically updates the configuration.

Configuring alert notifications, no UI


You learn to use the Kubernetes configuration method of configuring alert notifications that appear in the user notification system. Configuring notifications using this method has the same effect as using the UI method. The notifications supplement alert information in charts on the Manage/Hive/Compaction Observability dashboard in Grafana.

About this task

In this task, you see how to funnel alerts into a user notification system using a Kubernetes configuration available from your Database Catalog.

Procedure

1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate an environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.

3. Click  Show Kubeconfig .
The **Kubeconfig for environment** modal is displayed.
4. Click Copy Kubeconfig and paste the configuration into a file, and save the file as a YAML file, for example, kube_env.yml.
5. On the command line of the cluster, set the KUBECONFIG environment variable to the path of the YAML file.

```
$ export KUBECONFIG=<the-absolute-path-to>/kube_env.yml
```

6. Download the alert configuration using a command to fetch the configuration into the alertmanager.yml local file.

```
$ kubectl \
  -n istio-system \
  get configmap \
  alertmanager \
  -o "jsonpath={.data['alertmanager\.yaml']}" > alertmanager.yml
```

```
global:
  resolve_timeout: 5m

route:
  group_by: ['alertname', 'namespace', 'service']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 3h
  receiver: notification_system

receivers:
- name: 'notification_system'

inhibit_rules:
- source_match:
    severity: 'error'
  target_match:
    severity: 'warning'
  equal: ['alertid', 'namespace', 'service']
```

7. Edit the alert configuration to configure the receiver object by following instructions in the [Prometheus AlertManager v.0.20.0 documentation](#).

For example, a simple WebHook configuration looks something like this:

```
global:
  resolve_timeout: 5m

route:
  group_by: ['alertname', 'namespace', 'service']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 3h
  receiver: notification_system

receivers:
- name: 'notification_system'
  webhook_configs:
  - url: https://co-alerting.prod.mycompany.com
    send_resolved: true

inhibit_rules:
- source_match:
    severity: 'error'
  target_match:
```

```
severity: 'warning'
equal: ['alertid', 'namespace', 'service']
```

For information about integration possibilities, see the [receiver section](#).

8. Check that your configuration conforms to YAML syntax.

9. Update the related configmap.

For example:

```
$ kubectl \
  -n istio-system \
  create configmap \
  alertmanager \
  --from-file=alertmanager.yml=./alertmanager.yml \
  --dry-run=client -o yaml \
  | kubectl apply -f -
```

The alertmanager recognizes the change and automatically updates the configuration.

Checking an alert notification configuration

You learn how to ensure your alert notification configuration is in effect. You fetch the configuration directly from the alertmanager and check that it is reachable.

Procedure

1. Port-forward the prometheus service.

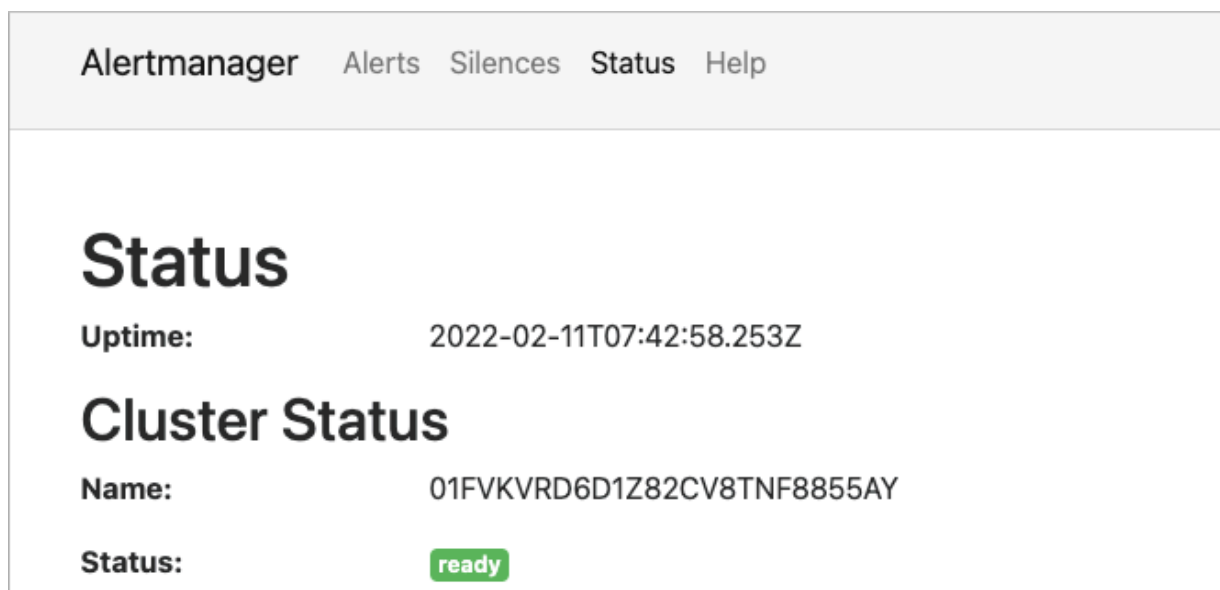
```
$ kubectl \
  -n istio-system \
  port-forward \
  service/prometheus \
  9093:9093
```

2. From another terminal the configuration is reachable by using the following curl command.

```
$ curl http://localhost:9093/api/v2/status
```

3. Check that the output shows your configuration.

Alternatively, to check the output, you can use the alertmanager WebUI by navigating to <http://localhost:9093/#/status>.



Deleting a compaction alert

Learn how to delete an user-defined or default compaction alert that you no longer require for your Cloudera Data Warehouse deployment.

Procedure

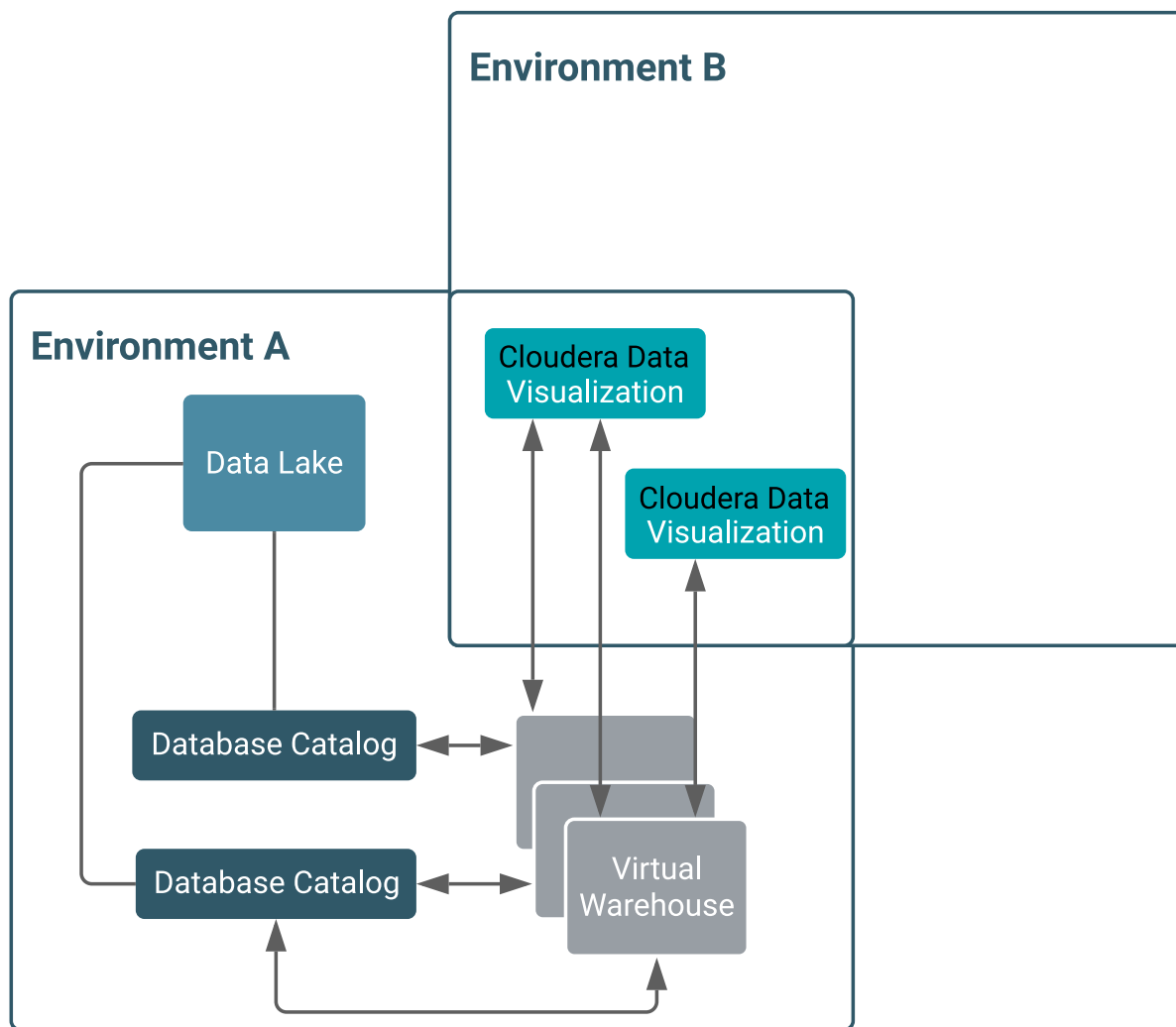
1. In the Cloudera Data Warehouse service, click Overview and go to the Environments tab.
2. Locate an environment that you activated for Cloudera Data Warehouse, which has the default Database Catalog.
3. Click Edit .
4. In the **Environment Details** page, click the ALERT SETTINGS tab, and then click Delete for the alert that you want to delete.
5. Click OK to confirm the delete action.

Data Visualization in Cloudera Data Warehouse

Cloudera Data Warehouse integrates Data Visualization for building graphic representations of data, dashboards, and visual applications based on Cloudera Data Warehouse data, or other data sources you connect to. You, and authorized users, can explore data across the entire Cloudera data lifecycle using graphics, such as pie charts and histograms. You arrange visuals on a dashboard for collaborative analysis.

You connect Data Visualization to a Virtual Warehouse as described in [Starting Data Visualization integrated in Cloudera Data Warehouse](#). Similar to using a BI client, you can configure and connect to Virtual Warehouses from different clusters. You configure the connection in a familiar way, providing an IP address or host name. Data Visualization is not tied to a particular Virtual Warehouse (VW). You can access data for your visualization from multiple Data Catalogs using multiple Hive or Impala Virtual Warehouses and multiple environments.

Kubernetes Cluster



Having multiple Data Visualization instances attached to an environment, you can create dashboards for different groups. For example, Marketing and Sales can have their own private dashboards. When you delete a Virtual Warehouse, your visuals remain intact.

Related Information

[Cloudera Data Visualization](#)

[Creating a visual](#)

[Working with datasets](#)

Rebuilding a Database Catalog

You can clean up resources and redeploy your Database Catalog while preserving the image version. Rebuild your Database Catalog to perform housekeeping or troubleshooting a problem.

About this task

Rebuilding the Database Catalog redeploys resources in the cluster while keeping the configurations and other settings you made using either the Cloudera Data Warehouse UI or CDP CLI.

You can rebuild the Database Catalog in the following ways:

- Using the [Beta version of Cloudera Data Warehouse CLI](#)
- Using the Rebuild option in the Cloudera Data Warehouse UI


Limitations

- Rebuilding deprecated versions of the Database Catalog is not supported and will fail.
- Rebuilding does not preserve the changes you made directly, using `kubectl` for example, to cluster resources in the affected namespaces.

Before you begin

- You must obtain the DWAdmin role.
- You must be running the version 2022.0.6-b92 or later of the Database Catalog to rebuild it.
- To preserve Hue documents, such as saved queries, you need to save a backup of the documents as described in *Backing up and restoring Hue documents*.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Database Catalogs tab.
3. Locate the Database Catalog you want to rebuild and click  Rebuild .
The **Rebuild Database Catalog** modal is displayed.
4. Click Rebuild Database Catalog.
5. Restore Hue documents as described *Backing up and restoring Hue documents*.

Results

The Database Catalog is rebuilt with the same image version as it had before.

What to do next

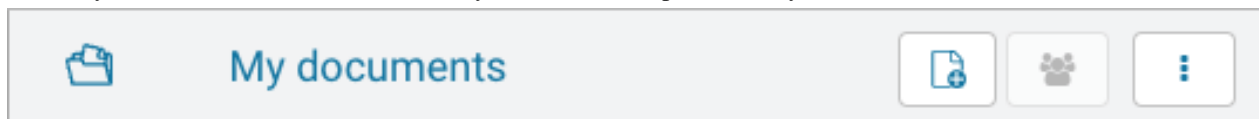
If you do not want to change JDBC/ODBC URLs, for example `env-xdd4x6.dw`, when rebuilding the Database Catalog, open a Support case to have the `CDW_CUSTOM_CLUSTER_ID` entitlement enabled.

Backing up and restoring Hue documents

You can back up and restore Hue data, such as a Hive or Impala query you saved in a JSON document, in the Hue database to prevent losing the documents after rebuilding the Database Catalog.

About this task

In Hue, you can view the JSON documents you created or imported in My documents.



Before you begin

- Avoid accessing or using the Hue web interface until the restore process is fully complete, as concurrent access may disrupt the operation and potentially lead to failure.

- During the Hue database restore operation it is critical to block any traffic to the running Hue services. If you cannot bring down the cluster, use the recommended workaround to [disable end user access](#) to the cluster endpoints. Failing to do so results in errors in addition to existing key constraints and other issues.

Procedure

1. Backup Hue documents you want to preserve from the command line.

```
# on hue pod
./build/env/bin/hue dumpdata -o /tmp/data.json
# on local machine
kubectl --kubeconfig ~/kconfig cp compute-1675582336-knmf/huebackend-0:/tmp/data.json -c hue ./data.json
```

2. Rebuild the Database Catalog as described in the previous topic.
3. Restore the Hue documents in the rebuilt Database Catalog.

```
#on local machine
kubectl --kubeconfig ~/kconfig cp ./data.json compute-1675582336-knmf/huebackend-0:/tmp/data.json -c hue

# on new hue pod
./build/env/bin/hue loaddata --ignorenonexistent /tmp/data.json
```

Rebuilding a Virtual Warehouse

You can clean up resources and redeploy your Virtual Warehouse while preserving its image version. Rebuild your Virtual Warehouse to perform housekeeping or troubleshooting a problem.

About this task

Rebuilding the Virtual Warehouse redeploys resources in the cluster while preserving the configurations and other settings you made using the Cloudera Data Warehouse UI or CDP CLI.

You can rebuild the Database Catalog in the following ways:

- Using the [Beta version of Cloudera Data Warehouse CLI](#)
- Using the Rebuild option in the Cloudera Data Warehouse UI


Limitations

- Rebuilding deprecated versions of the Virtual Warehouse is not supported and will fail.
- Rebuilding does not preserve the changes you made directly, using kubectl for example, to cluster resources in the affected namespaces.

Before you begin

- You must obtain the DWAdmin role.
- You must be running the version 2022.0.6-b92 or later of the Virtual Warehouse to rebuild it.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the Virtual Warehouses tab.
3. Locate the Virtual Warehouse you want to rebuild and click  Rebuild . The **Rebuild Virtual Warehouse** modal is displayed.

4. Click Rebuild Virtual Warehouse.

Results

The Virtual Warehouse is rebuilt with the same image version and configurations as it had before.

Upgrading Database Catalogs and Virtual Warehouses in Cloudera Data Warehouse on cloud

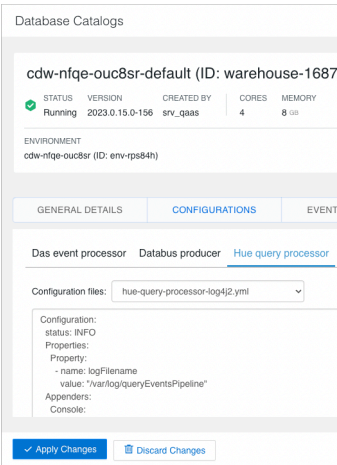
You need to learn about the choices, requirements, and alerts for upgrading Database Catalogs and Virtual Warehouses in Cloudera Data Warehouse on cloud.

Before you begin the upgrade

- Obtain the DWAdmin role.
- Take a look at the [release notes](#) for the following information:
 - Version mapping
 - What's New
 - Known issues
 - Fixed issues
 - Behavior changes
 - Older releases

Text and JSON configuration files

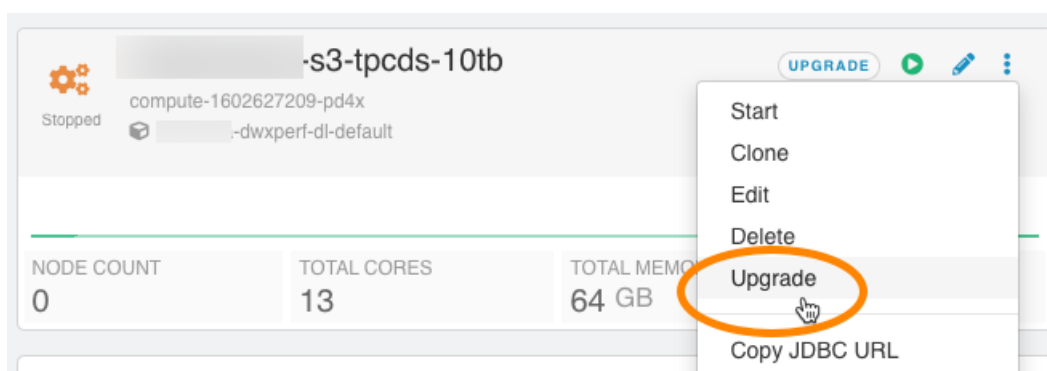
After upgrading, text and JSON configuration files in your upgraded Database Catalog and Virtual Warehouse are not upgraded. The pre-upgrade configurations are carried over to the upgraded Database Catalog and Virtual Warehouse. For example, in the Database Catalog Configurations , the .yaml file example below is not upgraded.



Configurations, such as key/value configurations, that are not in a text or JSON file are not affected by this limitation. These configurations are upgraded.

Upgrade available indicator

Database Catalogs and Virtual Warehouse must be compatible. The Cloudera Data Warehouse upgrade framework understands the interoperability constraints between them, and generates alerts. Alerts appear to indicate when upgrades are available for the Database Catalog and Virtual Warehouse as shown in the following image:



Latest releases versus compatible releases

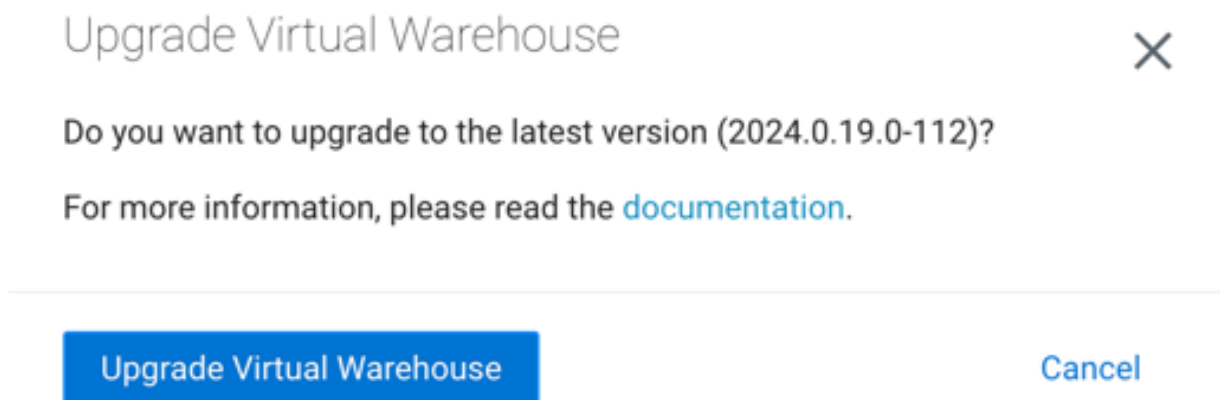
A release is considered *compatible* with the current release if its dependents or dependencies do not need to be upgraded. Compatible Database Catalogs do not need to be upgraded to function effectively with associated Virtual Warehouses. Compatible Virtual Warehouses do not need to be upgraded to function effectively with the associated Database Catalog.

Some releases contain changes that require upgrading associated components. For example, a Hive Virtual Warehouse might use new APIs introduced in a new release of the Hive metastore/Database Catalog. In this case, you must upgrade the Database Catalog that is associated with this Hive Virtual Warehouse. On the other hand, if you upgrade a Database Catalog, you are not required to upgrade its associated Virtual Warehouse. The Latest releases indicator appears in this case.

Available upgrade choices

When you click Upgrade, you might see one of the following options:

- Already up-to-date, which means there are no new releases that you can upgrade to.
- Upgrade to the latest version:



Click Upgrade Virtual Warehouse to start the upgrade process. The status of the upgrade appears: Upgrading, Starting, or Started.

Difference between upgrade and rebuild

Starting with the Cloudera Data Warehouse 1.8.5-b35 release, upgrade and rebuild operations perform different tasks. Upgrade operation updates the image version of the Cloudera Data Warehouse entity. Rebuild operation deletes and recreates the pods while preserving the image version and configurations.

Debugging Impala Virtual Warehouses using Web UIs

You can use the Catalog Web UI, Coordinator Web UI, and the StateStore Web UI to debug Impala Virtual Warehouses in Cloudera Data Warehouse.

About this task

The Impala daemons (impalad, statestored, and catalogd) debug Web UIs, which can be used in by using , is also available in the service. In service, the following Web UIs are provided:

- Impala Catalog Web UI

This UI provides the same type of information as the Catalog Server Web UI in . It includes information about the objects managed by the Impala Virtual Warehouse. For more information about this debug Web UI, see [Debug Web UI for Catalog Server](#).

- Impala Coordinator Web UI

This UI provides the same type of information as the Impala Daemon Web UI in . It includes information about configuration settings, running and completed queries, and associated performance and resource usage for queries. For information about this debug Web UI, see [Debug Web UI for Impala Daemon](#).

- Impala StateStore Web UI

This UI provides the same type of information as the StateStore Web UI in . It includes information about memory usage, configuration settings, and ongoing health checks that are performed by the Impala statestored daemon. For information about this debug Web UI, see [Debug Web UI for StateStore](#).

- Impala Autoscaler Web UI


This UI gives you insight into autoscaler operations (regular as well as workload-aware autoscaling), accessing log messages, and resetting the log level. The autoscaler Web UI includes information about the queries queued and running, executor groups, suspended calls, scale up/down calls, the autoscaler config, and the autoscaler logs.

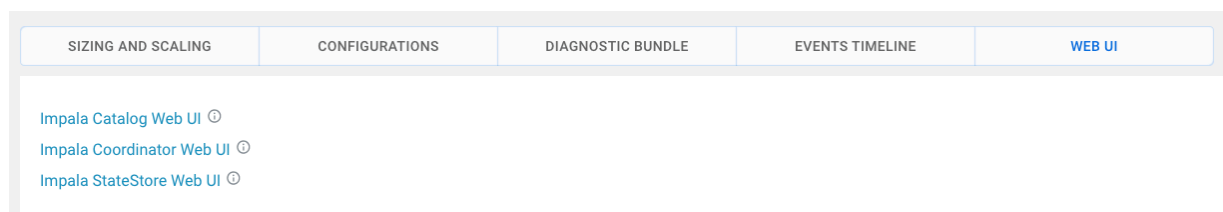
Required role: EnvironmentAdmin

Before you begin

Make sure that you note your workload user name and have set a password for it in the User Management module of . You need to use your workload user name and its associated password to log into the debug Web UIs. For more information, see [Setting the workload password](#) in the documentation set.

Procedure

1. In the UI on the Overview page, locate the Impala Virtual Warehouse for which you want to view the debug UIs, and select  Edit . The **Virtual Warehouse Details** page is displayed.
2. In the **Virtual Warehouse Details** page, select the Web UI tab. The list of debug Web UI links are displayed as shown in the following image:



3. Click a Web UI link corresponding to an Impala daemon that you want to debug. You are prompted to enter your workload user name and password.

Results

After you are authenticated, you can view the debug Web UI and use the information to help you troubleshoot issues with your Impala Virtual Warehouse.