Cloudera Data Warehouse on premises 1.5.5

Managing Virtual Warehouses

Date published: 2020-08-17
Date modified: 2025-11-08

https://docs.cloudera.com/

Legal Notice

Contents

Adding a new Virtual Warehouse

A Virtual Warehouse is an instance of compute resources that is equivalent to a cluster. It provides access to the data in tables and views in the data lake that correlates to a specific Database Catalog. Learn how to create a Virtual Warehouse in Cloudera Data Warehouse on premises.

About this task

Virtual Warehouses can only lookup the Database Catalog that they have been configured to access.

If you are evaluating Cloudera Data Warehouse or just learning, you can accept the default values. This task shows you how to create a Virtual Warehouse with the bare minimum configurations. You can create Virtual Warehouses with the following SQL engines in Cloudera Data Warehouse:

- Hive
- Impala
- Trino

Note: Trino is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Virtual Warehouses can use Hive, Impala, or Trino 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, and Trino is used to query large data sets distributed over one or more heterogeneous data sources.

Before you begin

Before you create a Virtual Warehouse in Cloudera Data Warehouse service, you must have completed the following tasks:

- Activate your environment for use in Cloudera Data Warehouse.

  Important: (On OpenShift environments) To activate an environment for the Cloudera Data Warehouse service, someone with adequate permissions must use the Red Hat OpenShift Local Storage Operator to create a local file system on an SSD/NVMe for each OpenShift worker node and then mount it to a known location on the worker node. This creates space for local caching.

  On Cloudera Embedded Container Service clusters, Cloudera Data Warehouse automatically creates the local file system. No additional steps are needed.
- Grant the DWAdmin role to the user or group that needs to create a Virtual Warehouse. This user or group determines which environment and Database Catalog your Virtual Warehouse uses.
- If your cluster is deployed on the OpenShift Container Platform (OCP) and you want to utilize autoscaling capabilities for your Trino Virtual Warehouse, you must install Kubernetes Event-driven Autoscaling (KEDA). KEDA enables dynamic scaling of containers, allowing them to scale up or down based on demand.

  Note: For Cloudera Embedded Container Service deployments, KEDA is automatically installed in the cluster.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. From the **Cloudera Data Warehouse Overview** page, click the **Virtual Warehouses** tab and click New Virtual Warehouse.
3. In the **Create Virtual Warehouse** drawer, specify a Name for your 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.

4. Select the Type (Hive, Impala, or Trino) of Virtual Warehouse.
5. Select the Environment and Database Catalog.
6. Select the Resource Pool that allows resources to be allocated for the Virtual Warehouse.
7. Select the Size of your Virtual Warehouse.
8. Specify settings for the Virtual Warehouse, including auto-suspend, auto-scaling, and query isolation, based on the type of Virtual Warehouse selected.
9. In the **User Groups** panel, select a user group to grant access to the Virtual Warehouse.
10. Select a Kubernetes Resource Template for the Virtual Warehouse.
11. Click Create Virtual Warehouse to create the new Virtual Warehouse.

Related Information

Hive Virtual Warehouse options
Impala Virtual Warehouse options
Trino Virtual Warehouse options
Resource templates for Cloudera Data Warehouse
How to size a Virtual Warehouse

Hive warehouse configuration options

While creating Hive Virtual Warehouses, you can set auto-suspend time, disable automatic suspension of the Virtual Warehouse, configure concurrency auto-scaling, and enable query isolation. These configurations help you in provisioning a high-performance and scalable Hive Virtual Warehouse.

Auto-suspend Virtual Warehouses

AutoSuspend Timeout is an option you can set while creating a Virtual Warehouse. Understand how the auto-suspend timeout option works along with the auto-scaling settings on a Virtual Warehouse.

What is auto-suspend

AutoSuspend Timeout 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.

How auto-suspend works

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.

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 (AutoSuspend Timeout), the Virtual Warehouse is suspended.

AutoSuspend Timeout sets the maximum time that the original warehouse executor group idles after all other executor groups have scaled down and released their resources. The JDBC endpoint is kept up and alive to keep the application connectivity and even respond to queries from the result cache or statistics where possible.

If a query is sent to a suspended Virtual Warehouse and if it cannot be answered from query result cache or statistics, then the query coordinator queues the query. When a queued query is detected, an executor group is immediately added to run the query on the Virtual Warehouse.

Note: When the auto-scaler increases or decreases the number of executor groups in the Virtual Warehouse, it can take up to several minutes before the scaling up or down takes effect. This slight delay might be caused by the time required to start new executors on the OpenShift cluster or for queries running on existing executors to finish. If the OpenShift cluster is at or near capacity, then scaling up does not occur until cluster resources are freed.

By default, auto-suspend is enabled. You can disable it by selecting the Disable Auto-suspend option.

Concurrency autoscaling

Concurrency autoscaling is an option you can set while creating a Hive Virtual Warehouse. Understand how concurrency autoscaling works and when to use it.

What is concurrency autoscaling

Concurrency Autoscaling is an option in Cloudera Data Warehouse that autoscales the Virtual Warehouse by adding additional executor groups when Hive receives concurrent queries. It is managed by the following components:

HiveServer

Receives all incoming queries and generates a preliminary query plan that does not include distributing the query across executor nodes. Then it looks for an available query coordinator to send the plan to executor nodes for distribution.

Query coordinators

Receives the serialized plan from HiveServer and generates the final query plan that distributes the query tasks across executor nodes for execution. There are as many query coordinators as executor nodes, but there is no one-to-one mapping. Instead, each query coordinator interacts only with the executor nodes in a group. For example, if you create a Virtual Warehouse that has 10 nodes (SMALL-size) and you scale up to 30 nodes by 10-node increments, a single query coordinator can only utilize the 10 executors from one group. They can never interact with executors from other groups. This supports effective isolation. Also it is important to note that one query coordinator can only orchestrate the execution of tasks for one query at a time, so the number of query coordinators determines your query parallelism limit.

Query executor containers

Can run multiple tasks of one or multiple different queries. Query executor nodes determine the throughput of your system. They are the executors which are the unit of sizing for Virtual Warehouses.

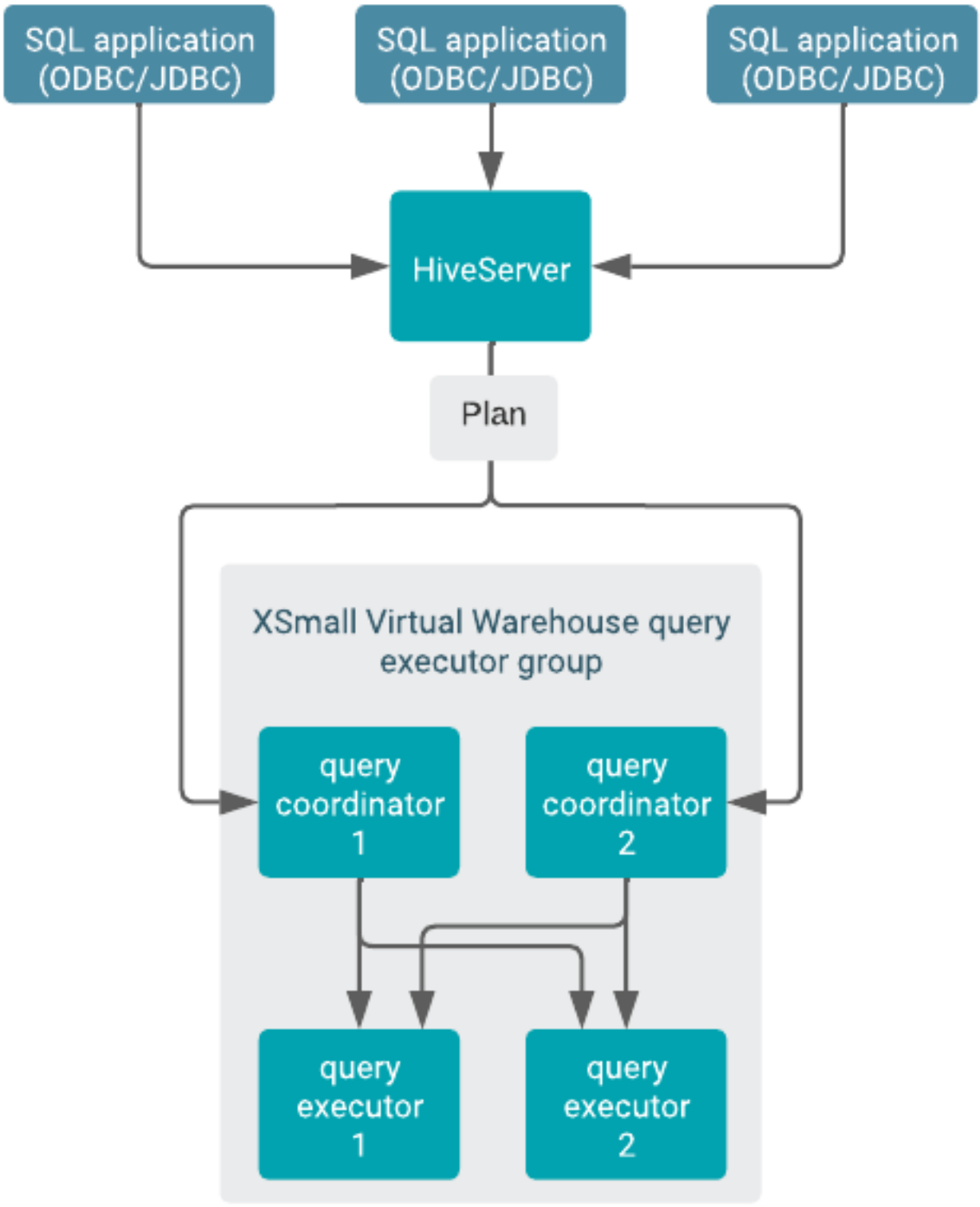
Executor groups

A group of executors that can run queries. The size of executor group is determined by the size you choose when you first create the Virtual Warehouse (XSMALL, SMALL, MEDIUM, or LARGE). The Virtual Warehouse size determines the maximum number of queries that each executor group can run concurrently. For example, if you create a SMALL Virtual Warehouse, each executor group can handle 10 parallel queries or tasks of up to 10 queries. A single query is always contained within a single executor group and never spans multiple executor groups. The throughput for an individual query is determined by the original size of the warehouse.

How concurrency autoscaling works

When the query load increases, auto-scaling increases. The query load grows proportionally with the number of concurrent queries and query complexity.

HiveServer locates an available query coordinator in the Virtual Warehouse to handle the query. The query coordinator generates the final query plan that distributes query tasks across available executors for execution. Each query coordinator can send query tasks to all query executors in the executor group of an XSmall Virtual Warehouse, as shown in the following image:

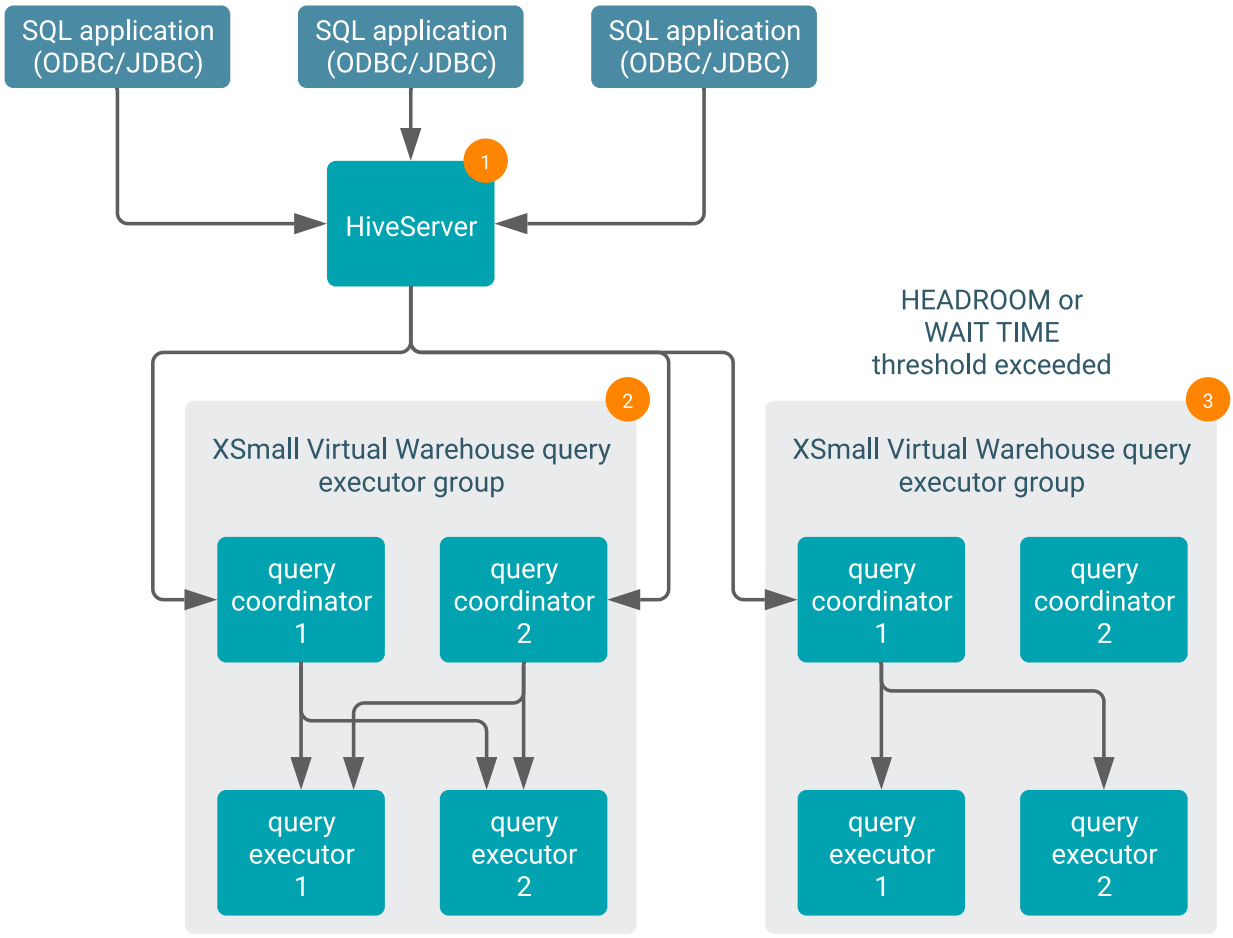


A single query on an idle Virtual Warehouse does not cause automatic scaling of resources. The query uses the resources available at the time of the query submission. Only subsequent queries can cause auto scaling. Additional query executor groups are added for scaling the resources.

The number of simultaneously running queries are equal to the number of query coordinators. A query executor can run 12 query fragments at the same time. The size of executor groups is determined by the size you choose when you create the Virtual Warehouse (XSMALL, SMALL, MEDIUM, or LARGE). A single query is always contained

within a single executor group and never spans multiple executor groups. The throughput for an individual query is determined by the original size of the warehouse.

Hive Virtual Warehouse auto-scaling manages resources based on query load:



1. When users connect to Hive Virtual Warehouses using SQL applications such as Hue or other SQL clients that use JDBC or ODBC, the query is handled by HiveServer. First HiveServer generates a logical plan that does not include distributing the query tasks across executor nodes.
2. Then HiveServer locates an available query coordinator in the Virtual Warehouse to handle the query. The query coordinator generates the physical query plan that distributes the query tasks across executor nodes for execution and locates available query executors to handle each query task. Each query coordinator can send query tasks to all query executors in the executor group and try to optimize for cache locality.
3. Auto-scaling occurs when either the headroom, or wait time, exceeds the threshold you configure. Additional query executor groups are added to perform the scaling.

There are as many query coordinators as executors, but there is no one-to-one mapping. Instead, each query coordinator can interact with all executors. However, one query coordinator can only orchestrate the execution of tasks for one query at a time, so the number of query coordinators determines the query parallelism limit.

Executors: The number of executors you need for a workload is analogous to the number of nodes needed for an on-premises workload.

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 do queries wait in the queue to execute? A query is queued if it arrives on HiveServer and no coordinator is available. For example, if Wait Time is set to 10, queries wait to execute in the queue for 10 seconds.

The warehouse auto scales up and adds an additional executor group. This ensures that your Virtual Warehouse does not take resources away from other workloads.

Note: Scaling might react to non-scalable factors to spin up clusters. For example, query wait times might increase because of inefficient queries and not because of query volume.

What concurrency autoscaling is best suited for

Concurrency auto-scaling is designed for running BI queries. 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

Query isolation

Query isolation is an option you can set while creating a Hive Virtual Warehouse. Understand how you can enable query isolation to configure the SQL engine to isolate queries that involve lengthy scanning, how to determine the default scan size threshold, and the queries types that benefit from query isolation.

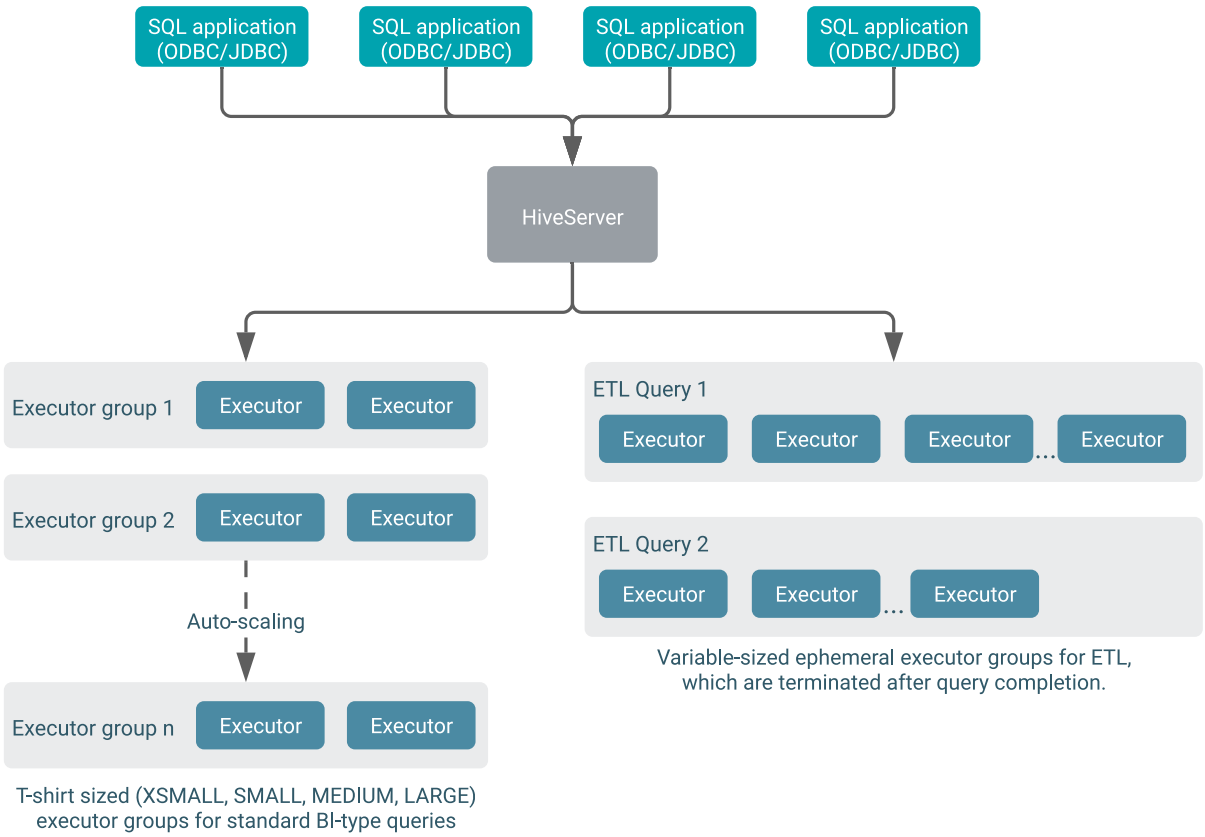
What is query isolation

Query Isolation enables you to isolate scan-heavy queries by dedicating compute resources to handle these scans, leaving other resources free to handle less intensive jobs, such as ad-hoc queries. When you enable query isolation, you can configure maximum concurrent isolated queries that a Hive Virtual Warehouse should support and maximum nodes that must be reserved for each isolated query. These parameters allow you to control resource usage and costs.

How query isolation works

While creating a Hive Virtual Warehouse, you can enable query isolation and select the value for Max Concurrent Isolated Queries, which corresponds to the hive.query.isolation.max.queries property. Any query that exceeds the hive .query.isolation.scan.size.threshold value runs in isolation. An isolated standalone Hive executor group is spawned to run the data-intensive, write-type queries if the estimated amount of data processed is above a specified threshold. Concurrency auto-scaling, auto-suspending, and auto-resume features of the core executor groups are not impacted by any isolated queries that are running.

After you specify the default scan size threshold, optimize your Virtual Warehouse using the controls described in Hive auto-scaling and enable query isolation for the Virtual Warehouse. When the query planner encounters a scan-heavy, data-intensive query whose scan size exceeds the value set for the threshold, it launches the query isolation feature. Query isolation automatically spawns an “on-demand” executor group with the estimated correct number of executors for the scan-heavy query.



When Analysts connect to Hive Virtual Warehouses using SQL applications such as Hue or other SQL clients that use JDBC or ODBC, the query is handled by HiveServer. First HiveServer generates a preliminary query execution plan that does not include distributing the query tasks across executors. The query planner also determines the data scan size of the query.

- For standard BI-type queries with smaller scan sizes: HiveServer locates an available query coordinator in the Virtual Warehouse to handle the query.
- For data-intensive queries with larger scan sizes: You can enable the query isolation feature and specify a scan size threshold when you are setting up your Virtual Warehouse. The query isolation feature spawns an on-demand executor group with the estimated correct number of executors to handle the single query. The executor group is limited in size by the values you specify for the query isolation parameters.

What query isolation is best suited for

Query isolation is designed for scan-heavy, data-intensive queries.

How query isolation is enabled

While creating or editing a Hive Virtual Warehouse, you can enable or disable query isolation and configure the following parameters:

Max Concurrent Isolated Queries (hive.query.isolation.max.queries)

Sets the maximum number of isolated queries that can run concurrently in their own dedicated executor nodes or the maximum number of queries that can spawn dedicated executor groups at one time. Select this number based on the scan size of the data for your average scan-heavy, data-intensive query. For example, if Max Concurrent Isolated Queries is set to 3 and a dedicated executor group is spawned for each data-intensive query, only 3 dedicated executor groups can be running at one time. If another data-intensive query is received, it must wait in a queue to run.

Max Nodes Per Isolated Query (hive.query.isolation.max.nodes.per.query)

Sets how many executor nodes can be created for each isolated data-intensive query.

How the default scan size threshold is determined

To determine the default scan size threshold (hive.query.isolation.scan.size.threshold), you need to calculate a threshold based on your Virtual Warehouse T-shirt size, and then set the default scan size parameter accordingly.

Multiply the executors in your Virtual Warehouse core executor group by the default data cache size. For example, for a MEDIUM-sized Virtual Warehouse, which has 20 executors, you set the scan size threshold to 20 x 200GB = 4TB.

Any query that scans more than 4TB of data runs in isolation, the number of executors spawned to run the query does not exceed the default setting for the hive.query.isolation.max.nodes.per.query parameter, which defaults to 2 times the T-shirt size. For a MEDIUM-sized Virtual Warehouse, which has 20 executors, 40 executors (2 x 20) are spawned to run the isolated query.

When the isolated query is running, if HiveServer receives another query, which scans less than 4 TB of data, the planner runs that query in the core executor group and does not spawn an isolated executor group to run it. If there is capacity in the core executor group, the query runs immediately or concurrency auto-scaling provisions more capacity in the core executor group.

Assuming the original data-intensive "scan-heavy" isolated query is still running, if HiveServer receives an additional "scan-heavy" query, which scans more than 4 TB of data, another isolated executor group is spawned to run the additional query. However, if two additional "scan-heavy" queries are received, unless the default value (2) is changed for the hive.query.isolation.max.queries parameter, one scan-heavy query runs, while the other is queued until execution of one of the other isolated queries is complete.

For more information about setting the default scan size threshold, see Tuning Hive Virtual Warehouses.

How the scan size threshold for Hive query isolation is set

Query Isolation enables your Virtual Warehouse to determine, based on the value you set for the hive.query.isolation .scan.size.threshold configuration parameter, whether to spawn dedicated nodes to run scan-heavy, data-intensive queries. You can set this threshold parameter in the **Virtual Warehouses** page as follows:

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to Virtual Warehouse ⋮ Edit CONFIGURATIONS Hiveserver2 and select hive-site from the Configuration files drop-down menu.
3. Specify the value for the hive.query.isolation.scan.size.threshold parameter in the VALUE text box in storage units. For example, 400 GB.
4. Click Apply Changes.

HiveServer2 High Availability in Cloudera Data Warehouse on premises

Cloudera Data Warehouse provides an option to enable active-passive configuration for HiveServer2 (HS2) pods in on premises for Hive and Unified Analytics. By selecting this option, two HS2 pods run simultaneously–one active and the other inactive. When one pod terminates, the inactive pod becomes active--most likey due to a node failure, providing High Availability (HA).

You can enable the Enable Active-Passive Hiveserver HA option while creating a new Hive or Unified Analytics Virtual Warehouse or editing it.

Important: Hive does not support session failover. Therefore, if an HS2 pod terminates for some reason, client sessions such as Hue, Beeline, JDBC, impala-shell, impyla, and so on may get disconnected until the inactive, standby pod becomes active.

Note: Enabling HS2 HA doubles the resource reservation–one for each HS2 pod.

Applying HS2 HA option on an existing Virtual Warehouse

1. Log in to the Cloudera Data Warehouse service.
2. Go to the Virtual Warehouse on which you want to enable HS2 HA, click ⋮ Edit , and go to the **Sizing and Scaling** tab.
3. Select the Enable Active-Passive Hiveserver HA option.
4. Click Apply Changes.

Impala warehouse configuration options

You can configure a number of parameters while configuring an Impala Virtual Warehouse in Cloudera Data Warehouse on premises. These parameters help you to provision high-performance Impala Virtual Warehouses and enable you to optimally utilize compute resources.

Workload Aware Auto-Scaling in Impala

Workload Aware Auto-Scaling (WAAS) allocates Impala Virtual Warehouse resources based on the workload that is running.

WAAS improves auto-scaling for the following types of workloads:

- Mixed or unknown workloads
- On-premises workloads you are migrating to Cloudera Data Warehouse
- Workloads running on a big cluster
- Workloads subject to queries with varying resource requirements
- Workloads you cannot easily segregate into different Virtual Warehouses
- Workloads clients must access from a single static endpoint

If you have already successfully segregated your workloads using multiple Virtual Warehouses, the benefits of enabling WAAS may be limited. The multiple Virtual Warehouse approach typically works well in this case.

Without WAAS, if you have mixed workloads, you size your Virtual Warehouse for the most resource-intensive queries, so the Virtual Warehouse can handle all your workloads.

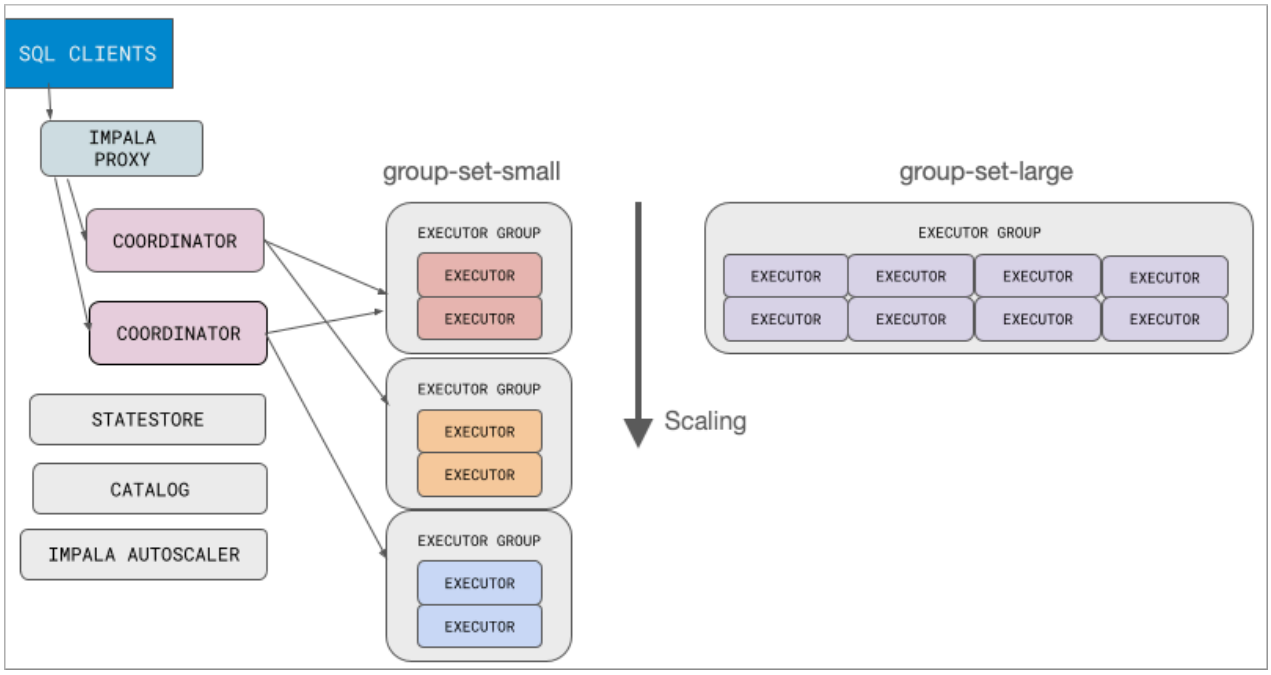
With WAAS, if you have mixed workloads, you define constraints for the overall warehouse size. You can add multiple sizes of executor group sets to spin up if needed by the workload. Within the executor group set, executor groups spin up or shut down as warranted by the workload. Each executor group set is mapped to an admission control resource pool, discussed later.

An executor group set is analogous to a range of nodes assigned to a Virtual Warehouse of a particular size. Dynamic provisioning of mixed workloads uses two, or more, executor group sets instead of a one-size-fits-all executor group.

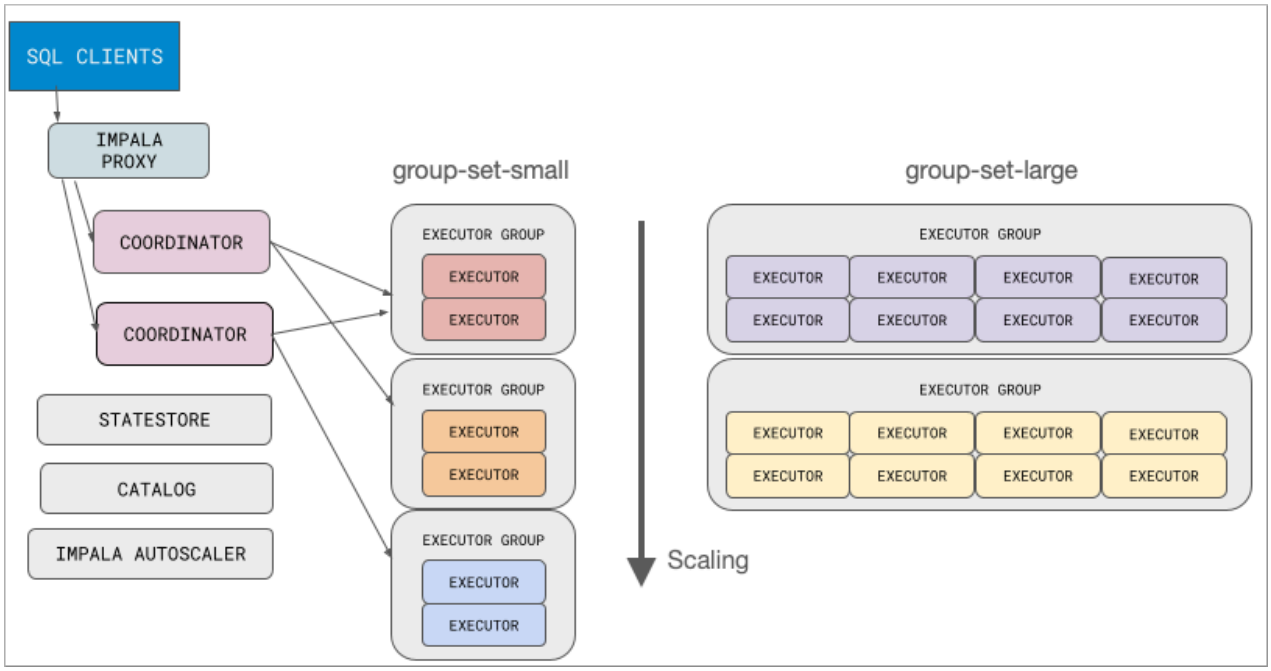
The following diagram depicts two executor groups sets in a WAAS Virtual Warehouse:

- group-set-small configured with an executor group size of 2 (small)
- group-set-large configured with an executor group size of 8 (large)

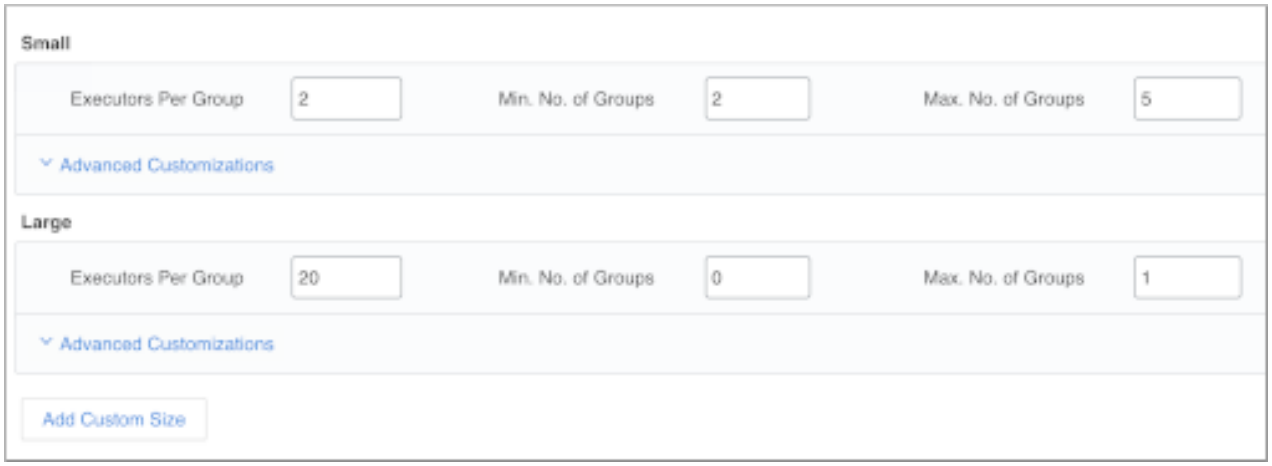
Within any executor group set are executor groups, all of the same size.



The following graphic depicts scaling that occurs when you run a large query. The executor group set is chosen based on the workload. The group-set-large spins up an executor group of size 8:



Using the UI shown below, you follow step-by-step instructions to configure a Small and Large executor group set.



You can expect the Small executor group set to handle the majority of cases, similar to a small (tee-shirt sized) Virtual Warehouse. This group keeps minimum resources for services. The Large executor group set can scale up greatly to service analytic queries. In Min No. of Groups, you configure a minimum number of groups to be allocated for processing queries before Impala auto-suspends. You configure a maximum number of groups based on your budget.

When migrating from a bare metal installation where your workload would run on hundreds of nodes, or more, consider also configuring an intermediate size using a Custom Set. You can add up to three Custom Sets for a total of five executor group sets. Cloudera recommends starting with no custom sets, establishing benchmarks, and adding one custom set at a time.

The following comparison highlights the differences in auto-scaling with and without WAAS.

Without WAAS	With WAAS
Executors provisioned when query runs for one-size-fits-all Virtual Warehouse	Executor group sets provisioned dynamically within range limits
Scaling occurs in multiples of the one-size-fits-all Virtual Warehouse	Scaling occurs within configured limits of two or more executor group sets
Small queries can run on the larger than necessary Virtual Warehouse	Small queries run on a smaller executor group set
Unused CPU expenses hard to control	Unused CPU expenses manageable
Large queries cannot run on a small Virtual Warehouse	Large queries run on an executor group of the appropriate size
SLA hard to manage	SLA manageable

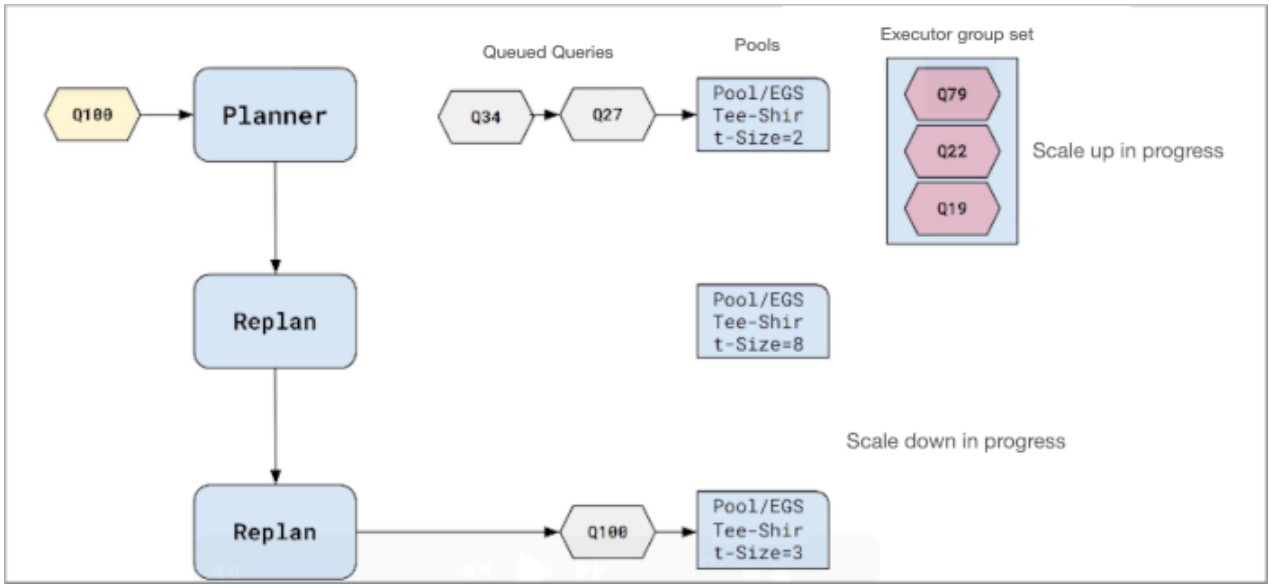
WAAS uses replanning to select an executor group set. See *How workload-aware autoscaling works*.

How workload-aware autoscaling works

The Impala planner traverses successively larger executor groups iteratively to find one that can run the query efficiently. Traversal stops when the planner finds an acceptable executor group set. In Workload Aware Auto-Scaling, this process is called replanning.

The decision regarding whether an executor group set can run the query is based on memory and CPU estimates. If the query will not fit into the estimated memory, or if the query needs more cores to run efficiently, the planner considers the next largest executor group set. If none is found, the query is queued to the largest pool.

The following diagram shows an example of the replan:



After replanning and establishing an executor group set able to run the query, the query enters the admission queue for a pool. The query will get admitted when the resources are available in the pool. The query may or may not succeed, depending on the resource settings and size of the executor group.

The size of the request pool and corresponding executor group set affects what queries can run there. No extra tuning is required. You can perform complex tuning of request pools using admission control by editing the xml configuration files. For more information, see Managing resources in Impala.

Workload Aware Auto-Scaling tuning

You can use an Impala query option to control query routing. You can also set a parameter to restrict queries from running in parallel under certain circumstances.

With Workload Aware Auto-Scaling (WAAS), Impala internally uses the REQUEST_POOL query option to control which executor group set is used for the query. WAAS is sophisticated, but if a query is being sent to the wrong pool, you can use the REQUEST_POOL query option to override the executor group set to be used.

Request pool query option

You can override Impala using the REQUEST_POOL query option to select a particular executor group set if you want. For example, high query latency might indicate the query is not being routed to the correct pool. You might get better performance from a larger pool than the REQUEST_POOL. Your large queries can be targeted at a specific pool using a request pool.

The following example overrides Impala using the REQUEST_POOL query option to route large queries to group-set-large:

```
set request_pool="group-set-large"
```

Parallelism query option

WAAS includes the capability in the Impala planner to reduce query parallelism under certain conditions.

The MT_DOP multi-threaded planning for queries in Cloudera Data Warehouse was previously static in nature. A query would use 12 threads per node and aggressively use all cores. In some cases, using all cores is unnecessary. A new feature can reduce query parallelism to more effectively use resources. Sophisticated planner functionality makes passes over the plan to determine how many cores are optimal for the query. Intelligence is gathered about fragments, how much work is involved in each fragment, and blocking operations in the plan.

Traversing fragments of the query plan tree, the planner propagates estimates of the max core count. Many details are gathered to determine the optimal parallelism for a pool. If parallelism is greater than the threshold (admission-

controller.pool-max-query-cpu-core-per-node-limit), traverse to the next larger pool. The information is analyzed and the planner restricts parallelism for fragments that require few threads. Costs are reduced.

To enable this feature, you restrict parallelism by setting the COMPUTE_PROCESSING_COST=1 query option.

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

- Obtain the entitlement for using the Workload Aware Auto-Scaling technical preview from your account team.
- You are familiar with the auto-scaling process and Workload Aware Auto-Scaling.
- 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.

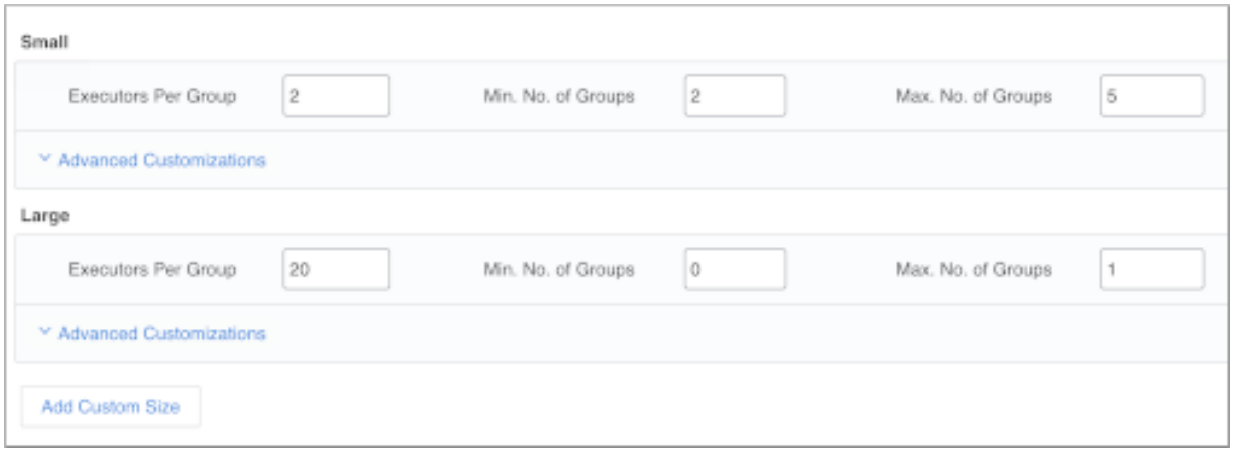
Procedure

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.



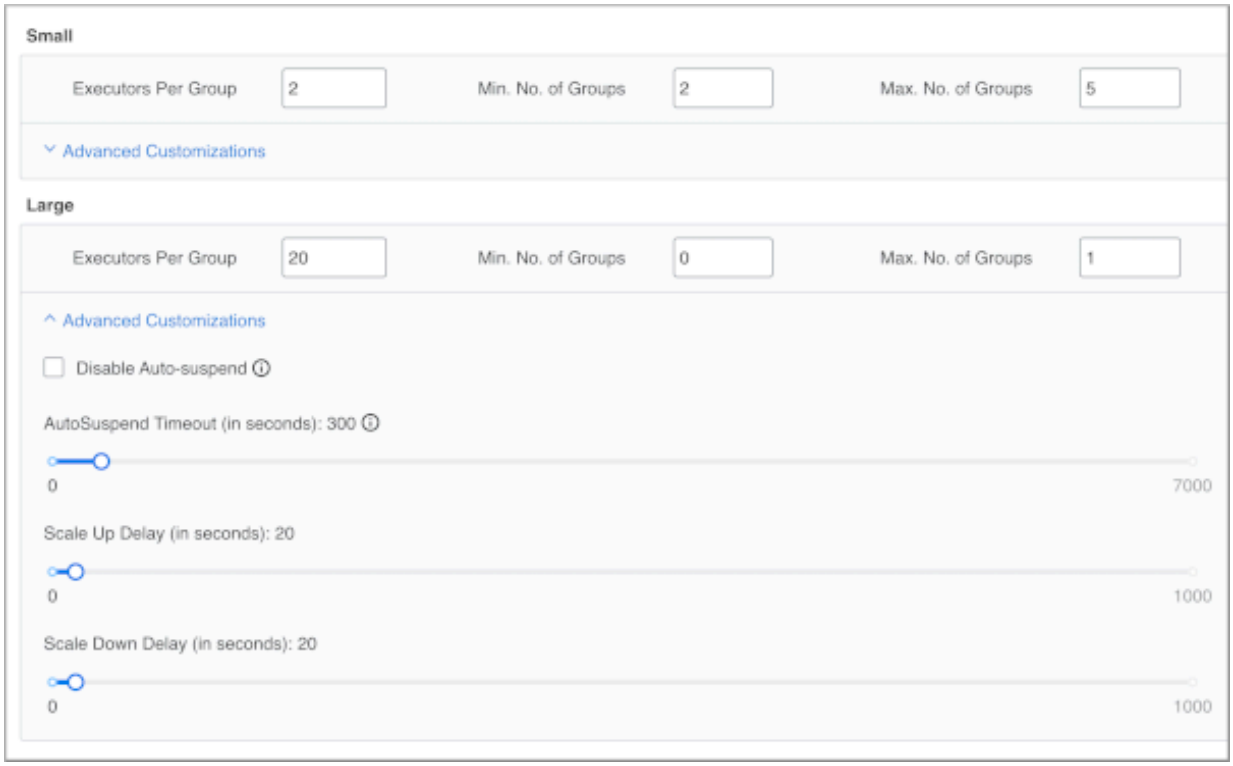
   An executor group set can contain up to 200 executors total. A group size (number of executors per group) must be within 1-100. The max no. of groups is limited by the group size . For example, a set with group size 2 can have up to 100 groups. A set with group size 100 can have up to 2 groups.
7. Optionally,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



   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.

Spill Impala queries to external storage

Cloudera Data Warehouse on on premises enables you to write intermediate files during large sorts, joins, aggregations, or analytic function operations to a remote scratch space on HDFS or Ozone.

What is spill to external storage

Spill to external storage is an option in Cloudera Data Warehouse where you can specify an HDFS or Ozone URI while creating an Impala Virtual Warehouse. This is where Impala writes intermediate files during large sorts, joins, aggregations, or analytic function operations.

How spilling to external storage works

To use this feature, you first configure the Impala daemon to use the specified locations for writing the intermediate files as described in Configuring Impala daemon to spill to HDFS. Then specify the HDFS or Ozone URI in the following format:

- HDFS:

```
hdfs://[***HOSTNAME***]:[***PORT***]/[***PATH***]:[***LIMIT***]
```

- Ozone:

```
ofs://[***SERVICE-NAME***]/[***PATH***]:[***LIMIT***]
```

Hostname and port are mandatory arguments that you must specify in the HDFS URI.

Note: When a valid HDFS URI is passed by the client, the 300G of local storage is used as a local disk buffer for spilling to HDFS.

Enabling Unified Analytics for Impala Virtual Warehouses in Cloudera Data Warehouse on premises

You can enable Unified Analytics while creating an Impala Virtual Warehouse. Doing this provisions a Virtual Warehouse that can automatically redirect queries to an appropriate SQL engine (either Hive or Impala) depending on the nature of the query.

Important: The Unified Analytics feature in the Cloudera Data Warehouse on premises is now deprecated. It is no longer supported in the user interface, but you can still create and manage Impala Unified Analytics Virtual Warehouses using the CDP CLI.

Why you should enable Unified Analytics

Unified Analytics is enabled by default when you create a Hive Virtual Warehouse. When you create an Impala Virtual Warehouse with the Unified Analytics option, you provision a hybrid entity with a common frontend, HiveServer2 (HS2), that takes and analyzes the queries submitted to it. HS2 compiles and submits the queries to the Hive or Impala SQL engines depending on whether the queries are reading or writing data.

- If the query is meant for reading data using Data Manipulation Language (DML) statements, specifically SELECT, then HS2 sends the query to Impala, favoring fast execution.
- If the query is meant for writing data (most Data Definition Language (DDL) queries, and INSERT INTO, INSERT OVERWRITE or CTAS (CREATE TABLE foo AS SELECT x FROM bar)-type queries), then HS2 sends the query to Hive, favoring reliable writes, resiliency, and a wider variety of formats that Hive can write to.

To enable Unified Analytics on an Impala Virtual Warehouse, turn on the Enable Unified Analytics option while creating an Impala Virtual Warehouse.

Additional Virtual Warehouse settings related to Unified Analytics

Because Hive-related services and components are added by Cloudera Data Warehouse when you create an Impala Virtual Warehouse with Unified Analytics, ensure that you review and configure the following additional options based on your needs:

Unified Analytics Authentication Mode

After you turn on Enable Unified Analytics option, select the authentication mode from the Unified Analytics Authentication Mode drop-down menu. The default authentication mode for the Hive components in the Unified Analytics mode is LDAP. The authentication mode that you set here applies only to the Unified Analytics' components (mainly Hive). The Impala components continue to support both LDAP and Kerberos authentication modes. If you connect to the Impala Virtual Warehouse remotely, then both LDAP and Kerberos authentication modes can be used. But if you

connect to the Impala Virtual Warehouse in the Unified Analytics mode, then only the selected authentication mode is used.

ETL Isolation

The ETL Isolation option is similar to the “Query Isolation” option that you can use for scan-heavy, data-intensive queries. The ETL Isolation option, while having the same meaning as "Query Isolation" for Hive, is a setting that is confined to the Hive executor groups created within the Impala Virtual Warehouse. It has no effect on the Impala coordinator or executor groups within the Impala Virtual Warehouse.

Max Concurrent Isolated Queries: Sets the maximum number of isolated queries that can run concurrently in their own dedicated executor nodes or the maximum number of queries that can spawn dedicated executor groups at one time. Select this number based on the scan size of the data for your average scan-heavy, data-intensive query. For example, if Max Concurrent Isolated Queries is set to 3 and a dedicated executor group is spawned for each data-intensive query, only 3 dedicated executor groups can be running at one time. If another data-intensive query is received, it must wait in a queue to run.

Max Nodes Per Isolated Query: Sets how many executor nodes that can be created for each isolated data-intensive query.

Enable Active-Passive Hiveserver HA

Option to enable active-passive configuration for HiveServer2 (HS2) pods in on premises for Hive and Unified Analytics. By selecting this option, two HS2 pods run simultaneously–one active and the other inactive. When one pod terminates, the inactive pod becomes active--most likey due to a node failure, providing High Availability (HA).

Related Information

Unified Analytics overview

Querying data in Unified Analytics

Authenticating users in Cloudera Data Warehouse on premises

Query isolation in Hive

Auto-suspend Virtual Warehouses

AutoSuspend Timeout is an option you can set while creating a Virtual Warehouse. Understand how the auto-suspend timeout option works along with the auto-scaling settings on a Virtual Warehouse.

What is auto-suspend

AutoSuspend Timeout 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.

How auto-suspend works

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.

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 (AutoSuspend Timeout), the Virtual Warehouse is suspended.

AutoSuspend Timeout sets the maximum time that the original warehouse executor group idles after all other executor groups have scaled down and released their resources. The JDBC endpoint is kept up and alive to keep the application connectivity and even respond to queries from the result cache or statistics where possible.

If a query is sent to a suspended Virtual Warehouse and if it cannot be answered from query result cache or statistics, then the query coordinator queues the query. When a queued query is detected, an executor group is immediately added to run the query on the Virtual Warehouse.

Note: When the auto-scaler increases or decreases the number of executor groups in the Virtual Warehouse, it can take up to several minutes before the scaling up or down takes effect. This slight delay might be caused by the time required to start new executors on the OpenShift cluster or for queries running on existing executors to finish. If the OpenShift cluster is at or near capacity, then scaling up does not occur until cluster resources are freed.

By default, auto-suspend is enabled. You can disable it by selecting the Disable Auto-suspend option.

Configuring Impala coordinator shutdown

To optimize resource utilization, 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

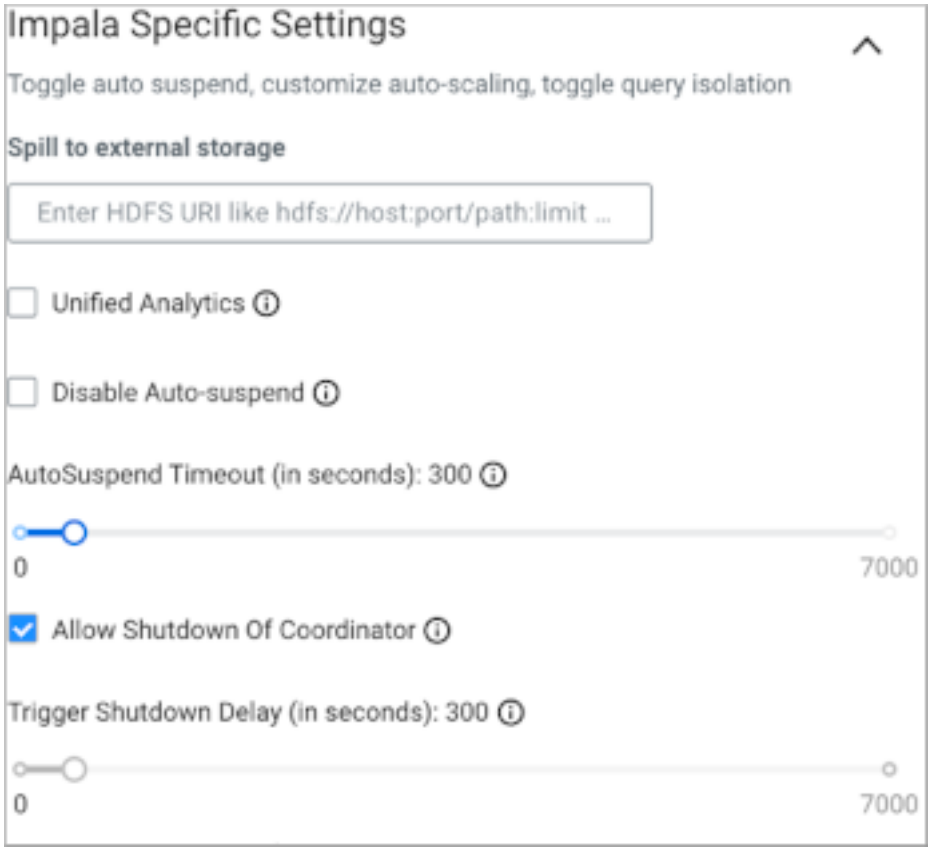
- Update impyla, jdbc, impala shell clients if used to connect to Impala.

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.

Auto-scaling Impala Virtual Warehouses

Your Impala Virtual Warehouse in Cloudera Data Warehouse Private Cloud has an auto-scaler process that works with coordinators and executors to make resources available for queued queries. This ensures that workload demand is met without wasting cloud resources.

How autoscaling in Impala works

How coorninators and executors are used to autoscale resources is explained as follows:

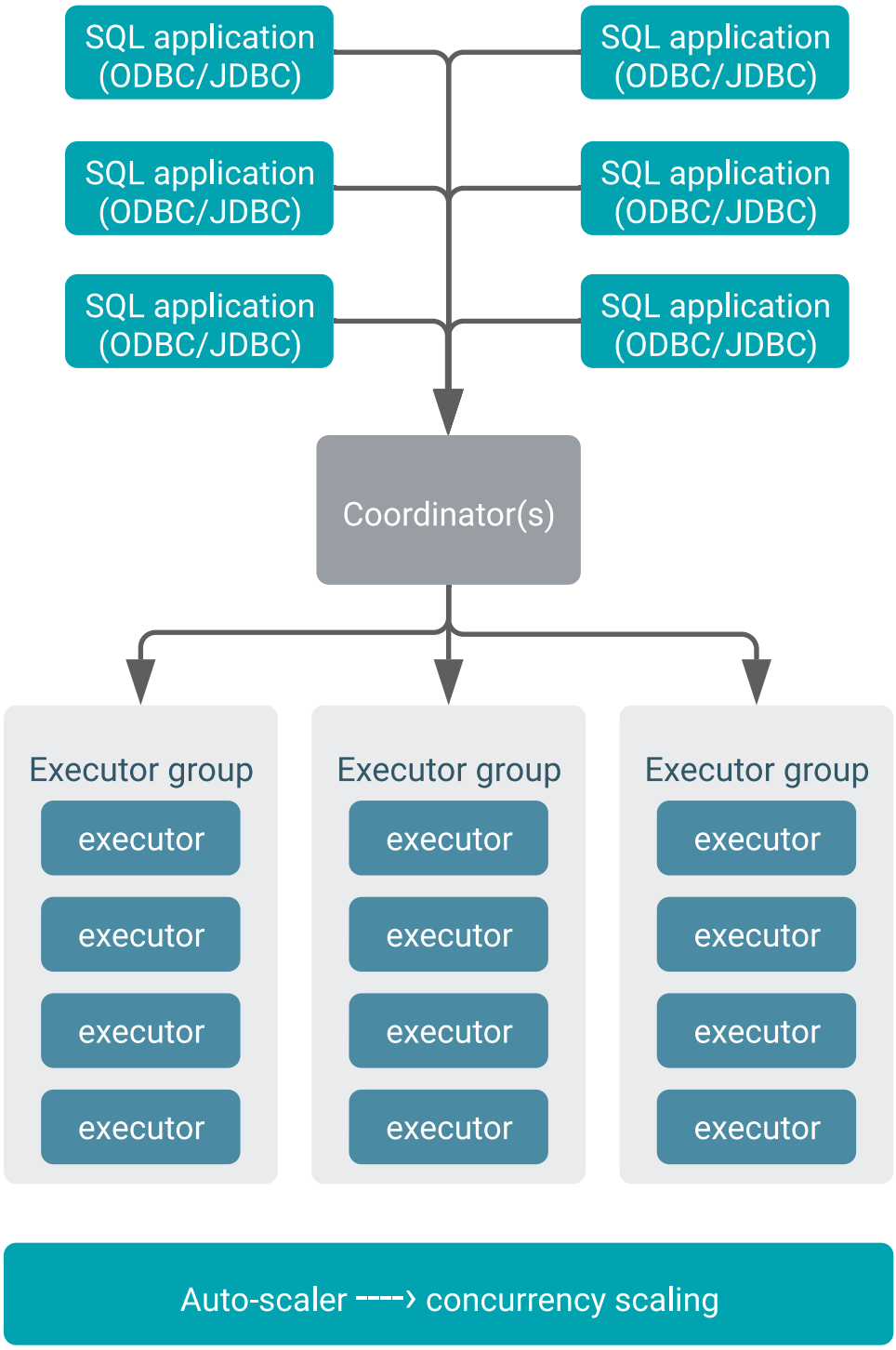
- Coordinator processes: Handle all incoming queries, creating execution plans and handing off the query to executor processes for execution. By default, there are two coordinator processes to enable high availability and fault tolerance. You can configure a single coordinator or an active-passive coordinator pair to resolve or mitigate query concurrency problems.. If one coordinator process fails, a backup coordinator process takes over so there is no single point of failure. By using only one coordinator process, you are likely to save on cloud resource consumption.
- Executor processes: Processes that execute query fragments. Executor processes run on executor nodes, the unit of sizing for Virtual Warehouses. Each executor node runs one executor.

- Executor groups: A group of executors that can execute queries. 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. The size of executor groups is determined by the size you choose when you first create the Virtual Warehouse (XSMALL, SMALL, MEDIUM, or LARGE).

  It is recommended that you select the size of a Virtual Warehouse based on the number of nodes you need to run a typical query for your workloads based on your normal query size and complexity. For example, if you expect that 20 nodes are needed to run a typical query in your workloads, you can create a medium-sized Virtual Warehouse, which by default has 20 executor nodes. However, if you have a large number of queries that must be run in your workloads concurrently, you can set the minimum executor node count to 40 and maximum count to 80 by using the Nodes: Min: Max: setting. Then, each executor group for this Virtual Warehouse contains the 20 executors required to run an average query, but you have two executor groups to handle the volume of queries in your workloads. If the total number of executor nodes auto-scales up to 80, then there will be 4 executor groups containing 20 executor nodes each.

  The rationale behind selecting a size based on the normal size and complexity of your queries is that then the Virtual Warehouse can run all your normal queries in an acceptable time. Selecting size based on this criteria prevents you from "over-sizing" your Virtual Warehouse and as a result, unnecessarily incurring extra costs. On the other hand, keep in mind that under-sizing your Virtual Warehouse might result in queries spilling, which makes them run slower, in spite of the fact that they may eventually complete correctly.
- Auto-scaler: A process that monitors Impala to determine when more or fewer resources are needed. When the auto-scaler detects an imbalance in resources, it sends a request to the Kubernetes framework to increase or decrease the number of executor groups in the Virtual Warehouse.

Impala uses memory-based auto-scaling to manage resources:



- Coordinator(s): When users connect to Impala Virtual Warehouses using SQL applications such as Hue, the Impala shell, or other SQL clients that use JDBC or ODBC, the query is handled by the coordinator process. First, the coordinator generates an execution plan for the query. The execution plan includes an estimate of the memory required to run the query.
- Executor group: Then the coordinator locates an executor group that has enough available memory to run the query. Each executor group is limited by the number of queries and the memory available to run them. Currently,

executor groups can handle up to 3 large queries. Executors can handle more queries that are simpler and that do not utilize concurrency on the executor. If there are no executor groups with enough available resources to handle a query, it is queued until resources become available.

- Auto-scaler: When the auto-scaler detects that there are queued queries, it adds executor groups to the Virtual Warehouse to execute the queries. The auto-scaler starts scaling up and down the number of executor groups according to what is set in the auto-scale settings:
  - 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 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 is affected by how the underlying Kubernetes system is configured.

When the auto-scaler has scaled back to the last executor group, which contains the default number of executors for the Virtual Warehouse, and those executors are idle, the Virtual Warehouse is suspended. You can override this behavior by selecting the Disable Auto Suspend option. The time waits, after there are no longer any queries running and before the Virtual Warehouse is auto-suspended, is determined by the value set for AutoSuspend Timeout. The default number of executors per executor group is based on the number of executor nodes contained by the original size of the Virtual Warehouse when it was created. For example, if the Virtual Warehouse was created as MEDIUM-sized, which has 20 executor nodes, then each executor group contains 20 executors.

If a query is executed on a suspended Virtual Warehouse, then the coordinator queues the query. When the auto-scaler detects a queued query, it immediately adds an executor group that can run the query on the Virtual Warehouse.

Note: 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.

About the Impala Autoscaling Dashboard

The Impala Autoscaling Dashboard helps you to visualize and understand how the Impala Virtual Warehouse autoscales, how the queries are routed to an executor group set, how the queries affect the provisioning of the executors, and resource utilization over a specified time window. You can use this dashboard to monitor the regular as well as workload-aware autoscaling.

Impala Autoscaling Dashboard: Key insights

The Impala Autoscaling Dashboard enables you to obtain insight about the inner workings of how an Impala Virtual Warehouse autoscales. It contains the following graphs for each executor group set:

Query Execution Timeline

Enables you to analyze to which executor group sets the Impala queries are routed and why

- You can visualize which executor group sets are assigned what queries. This mainly depends on the resources available to the executor group sets.
- You can also view the timeline of each query's execution: how long was the query queued, when was it admitted, how long did it take to run, and so on.

Compute Metrics

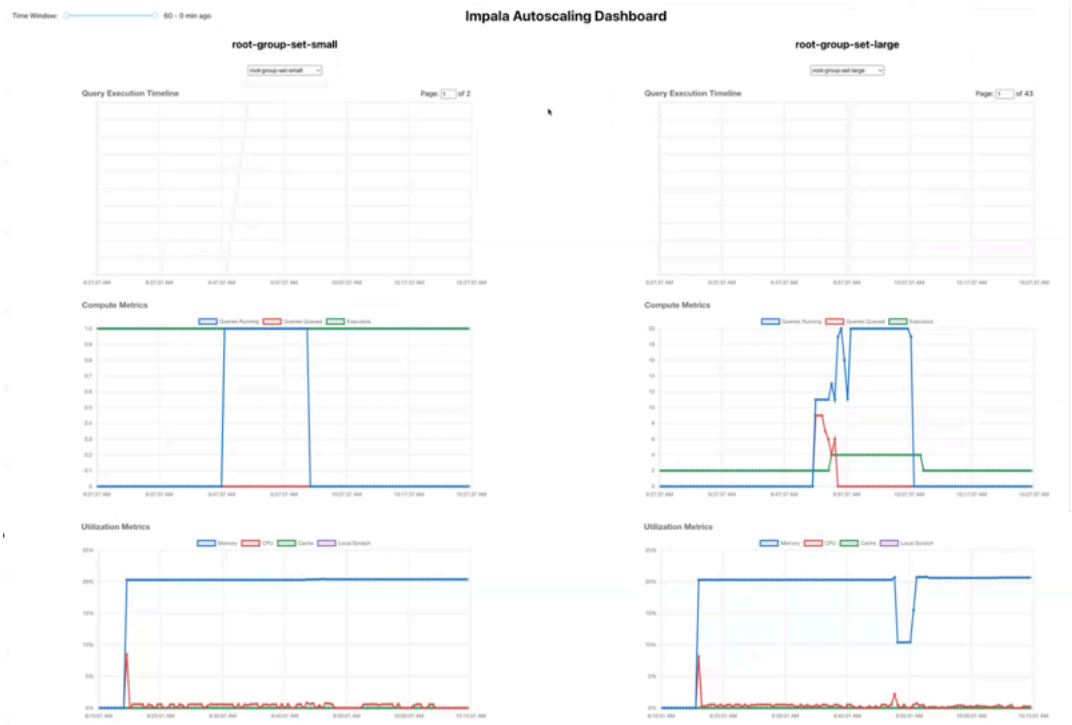
Enables you to analyze how the queries that are queued and run affect the provisioning of the executors

- You can view the number of queued or running queries at any given time.
- You can also view the number of executors that are running, or have scaled up or scaled down over a specified time window.

Utilization Metrics

Enables you to analyze resource utilization of the running the queries

- You can view the percentage of available CPU, memory, scratch, and cache that is being used.



Impala Autoscaling Dashboard: Layout and controls

Each column on the Impala Autoscaling Dashboard is dedicated to an executor group set or a resource pool. For each executor group set, you get interactive charts to analyze query execution timeline, compute metrics, and utilization metrics. Columns for new executor group sets are added if you have selected to use additional executor group sets. If you have not enabled workload-aware autoscaling on your Impala Virtual Warehouse, then only one executor group set is deployed, and you see only one column on the dashboard.

You can use the Time Window slider to zoom into a specific time period.

Under each executor group set, there is a drop-down menu that enables you to select executor groups with a set. It enables you to view and analyze the metrics for a specific executor group in detail.

Each query execution timeline page is dedicated to a set of queries that has run in a recent time period. The first page shows the most recently run queries. You can navigate through older queries by increasing the Page number.

On the Query Execution Timeline graph, you can view all the phases of the query's lifecycle. To view details such as query ID, start and end time of a phase, the reason why this query was assigned to this executor group set, you can hover over a phase. You can view the full query profile by clicking on the query's phase bar.

Accessing the Impala Autoscaling Dashboard

You access the Impala Autoscaling Dashboard from the Web UI tab of the Virtual Warehouse Details page in Cloudera Data Warehouse.

Procedure

1. Log in to the Cloudera Data Warehouse service as an EnvironmentAdmin.
2. Go the Virtual Warehouses tab and click ⋮ Edit corresponding to your Impala Virtual Warehouse.
   The **Virtual Warehouse Details** page is displayed.

3. Go to the Web UI tab and click Impala Autoscaler Web UI.
   The Impala Autoscaling Dashboard opens in a new tab.
   Specify your workload credentials to authenticate.

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).

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 Runtime component versions for Cloudera Data Warehouse on premises.
   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 or later.

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 1: HTTPAuthCookies

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

Configuring fe_service_threads in Cloudera Data Warehouse on premises

The “fe_service_threads” configuration is used to specify the maximum number of concurrent client connections or threads allowed to serve client requests in an Impala Virtual Warehouse. The default value of the “fe_service_threads” configuration is 128. You can change the value of this configuration from the Impala Coordinator flagfile configurations in the Cloudera Data Warehouse UI.

About this task

The recommended value of the “fe_service_threads” configuration is 128. Setting the value of this configuration lower than 128 can degrade performance.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the **Virtual Warehouses** tab and locate the Impala Virtual Warehouse on which you want to set this configuration.
3. Click ⋮ Edit .
4. Go to CONFIGURATIONS Impala coordinator and select flagfile from the Configuration files drop-down menu.
5. Specify the value of the “fe_service_threads” configuration in the VALUE field.
6. Click Apply Changes.

Trino warehouse configuration options

When creating Trino Virtual Warehouses, you can enable auto-shutdown for Trino worker pods. This feature helps optimize resource utilization and ensures the provisioning of a high-performance Trino Virtual Warehouse.

Note: Trino is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Configuring auto-suspend for Trino worker pods

Learn how to configure Trino worker pods to automatically shutdown during idle periods, which allows you to optimize the resource utilization.

Before you begin

If your cluster is deployed on the OpenShift Container Platform (OCP) and you want to utilize auto-suspend capabilities, you must install Kubernetes Event-driven Autoscaling (KEDA). KEDA enables dynamic scaling of containers, allowing them to scale up or down based on demand.

Note: For Cloudera Embedded Container Service deployments, KEDA is automatically installed in the cluster.

Procedure

1. Follow the instructions for "Creating a Virtual Warehouse".
2. Select a size for the Virtual Warehouse.
3. Select the Enable Auto Suspend checkbox to allow Trino worker pods to automatically shut down during idle periods. You can also specify a timeout period (in seconds) for the auto-suspend.

   The timeout period defines the duration that KEDA waits before scaling down worker nodes to zero. By default, this value is set to 300 seconds.
4. Click Create Virtual Warehouse.

Related Information

Installing KEDA for OCP

Installing KEDA for OpenShift Container Platform

Learn how to install the Kubernetes Event-driven Autoscaling (KEDA) on a Redhat Openshift Container Platform cluster using Helm.

About this task

KEDA can be installed on any Kubernetes cluster and helps to scale up or scale down the container to ensure resources are available for the workload.

Important: Only one instance of KEDA can be run on a single Kubernetes cluster. Cloudera recommends installing KEDA version 2.17.1 or 2.17.2.

Procedure

1. Log in to the Cloudera Management Console and go to Environments Compute Cluster .
2. In the **Compute Cluster** tab, click Actions Download Kubernetes Configuration to download the kubeconfig.txt file to your system.
3. Set the KUBECONFIG environment variable to the path of the downloaded file.

   ```
   export KUBECONFIG=[***kubeconfig.txt file path***]
   ```
4. Add the KEDA Helm repository and update the Helm repository to fetch the latest chart information.

   ```
   helm repo add kedacore https://kedacore.github.io/charts
   heml repo update
   ```
5. Create a dedicated namespace for KEDA. If the keda namespace does not already exist, create the namespace and add the label, monitoring-platform-access=true.

   ```
   kubectl create namespace keda
   kubectl label namespace keda monitoring-platform-access=true
   ```
6. Install KEDA by specifying the required version, for example, 2.17.2, and the namespace that you created in the previous step.

   ```
   helm install keda kedacore/keda --version 2.17.2 --namespace keda
   ```

   The command deploys KEDA and its associated resources, including Custom Resource Definitions (CRDs), into the Kubernetes cluster under the keda namespace.

7. Check the keda namespace for running pods to verify a successful deployment.

```
kubectl get pods -n keda
```

Related Information
KEDA concepts

Disabling metadata synchronization

By default, metadata is synchronized when refreshed/invalided 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.

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.

Auto-scale threshold settings

This topic provides information about the auto-scaling threshold settings for Hive and Impala Virtual Warehouses in Cloudera Data Warehouse on premises.

When you create new Virtual Warehouse instances, you can set auto-scaling thresholds. These thresholds set limits on automatic cluster scaling to meet workload demands. Setting these limits prevents warehouses from consuming too many resources when workload demands increase or decrease. Another important benefit of enabling auto-scaling for

your Virtual Warehouse is that it further enforces node isolation, increasing warehouse fault tolerance. You can adjust the following auto-scaling thresholds:

Hive-LLAP Virtual Warehouse auto-scaling threshold settings

The following settings are available to configure auto-scaling for Hive-LLAP Virtual Warehouses:

Hive-LLAP Auto-scaling Threshold	Description
AutoSuspend Timeout	Sets the maximum time the warehouse idles before shutting down.
Nodes: Min: <n>, Max: <n>	Sets the minimum and maximum number of nodes (executors) for the warehouse cluster. The maximum number of executors is limited by your cloud account limits. Choose the minimum and maximum number of executors based on two factors: • Average number of queries that must be run concurrently for your workloads. The more queries that must be run concurrently, the larger number of executors are needed. • The size of the data your workloads access. Larger numbers of executors can cache more data, which enhances performance.
WAIT TIME	Sets how long queries wait in the queue to execute. For example, if WaitTime Seconds is set to 10, then when executing queries are waiting in the queue for 10 seconds, the cluster auto-scales up to meet query demand.
Query Isolation	Enables the Virtual Warehouse to determine, based on the value you set for the hive.query.isolation.scan.size.threshold configuration parameter, whether to spawn dedicated executor nodes to execute scan-heavy, data-intensive queries in isolation.
Max Concurrent Isolated Queries	Available if Query Isolation is enabled. Sets the maximum number of isolated queries that can execute 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.
Max Nodes Per Isolated Query	Available if Query Isolation is enabled. Sets how many executor nodes can be spawned for each isolated query.

Impala Virtual Warehouse auto-scaling threshold settings

The following settings are available to configure auto-scaling for Impala Virtual Warehouses:

Impala Auto-scaling Setting	Description
Disable AutoSuspend	When you enable this control, your Virtual Warehouse does not suspend itself when the auto-scaler has scaled back to the last executor group, and those executors are idle. Instead, the Virtual Warehouse continues to consume cloud resources. You can override this behavior by disabling the Disable AutoSuspend control.
AutoSuspend Timeout	Sets the maximum time the warehouse idles before shutting down. This setting only applies when the Disable AutoSuspend control is not enabled.
Nodes: Min: <*N*> Max: <*N*>	Sets the minimum and maximum number of nodes (executors) for the warehouse cluster. The maximum number of executors is limited by your cloud account limits. Choose the minimum and maximum number of executors based on two factors: • Average number of queries that must be run concurrently for your workloads. The more queries that must be run concurrently, the larger number of executors is needed. • The size of the data your workloads access. Larger numbers of executors can cache more data, which enhances performance.
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 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.

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.

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'
use workload aware autoscaling	Set your smallest and largest executor group sizes. Custom sets can be added with executor group sizes inside this range (3 Maximum). All group sizes must be within 1 and 100.

If you are evaluating , 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.

Correcting the Virtual Warehouse size

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. 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 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 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, Impala, or Trino.
   - 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.

Creating Impala tables in Kudu in Cloudera Data Warehouse on premises

Cloudera Data Warehouse allows you to create Impala tables in Kudu. You can configure an Impala Virtual Warehouse to connect to Kudu and create Impala tables in Kudu using Hue. Or, you can create tables on the fly by specifying the Kudu master host in the TBLPROPERTIES statement while running the query from the Hue query editor.

About this task

Attention: This feature is in technical preview and not recommended for use in production environments. Cloudera recommends that you try this feature in test or development environments.

Before you begin

Obtain the hostname of the Kudu master home by going to Cloudera Manager Clusters Kudu service Instances from the Cloudera Management Console.

Creating Impala tables in Kudu on the fly

To create Impala tables in Kudu without updating a Virtual Warehouse's Impala coordinator configuration, you must specify the Kudu master host in the TBLPROPERTIES statement as follows while running the query from Hue:

```
TBLPROPERTIES ('kudu.master_addresses'='[***host.example.com***]')
```

Configuring the Virtual Warehouse to create Impala tables in Kudu

By reconfiguring an existing Impala Virtual Warehouse as follows, any tables you create will be created in Kudu.

Procedure

1. Log in to the Cloudera Data Warehouse service as a DWAdmin.
2. Go to an Impala Virtual Warehouse and click ⋮ Edit Configuration Impala coordinator and select flagfile from the drop-down list.
3. Click + and enter the following key and value:

Key	Value
kudu_master_hosts	*[***HOSTNAME-OF-KUDU-MASTER***]*

4. Click Apply Changes.
5. Restart the Virtual Warehouse.
6. Open Hue from the same Virtual Warehouse.
7. Enter the following lines in the query editor and click the run button:

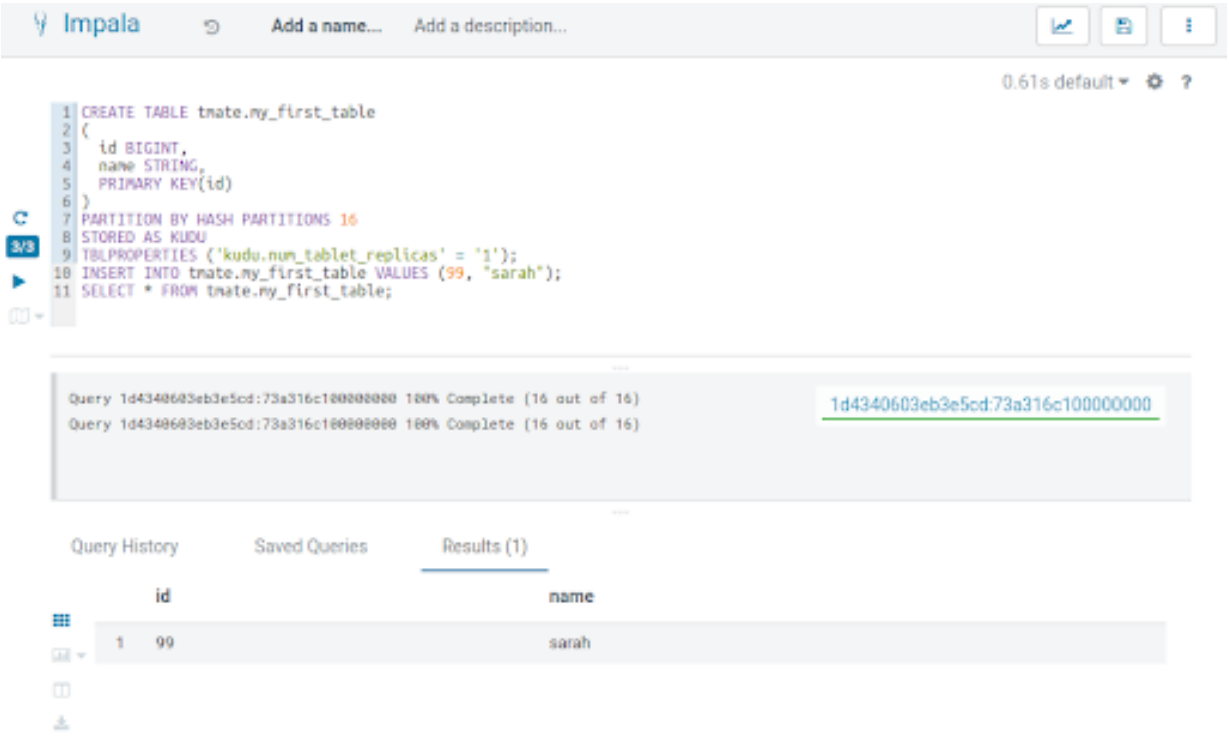
```
-- Create a new table
-- Use the kudu.num_tablet_replicas if the Kudu cluster is too small
CREATE TABLE my_first_table
(
  id BIGINT,
  name STRING,
  PRIMARY KEY(id)
)
PARTITION BY HASH PARTITIONS 16
STORED AS KUDU
TBLPROPERTIES ('kudu.num_tablet_replicas' = '1');

-- Insert into Kudu table
INSERT INTO my_first_table VALUES (99, "sarah");

-- Verify if the data was inserted
```

```
SELECT * FROM my_first_table;
```

The above commands create an Impala table in Kudu and insert a sample record. The following is a screenshot showing the SQL commands and their output in Hue:



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.

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 premises. They support the data modifications made as a result of ACID transactions.

Initiator

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.

Important: For the default Database Catalog, which is the Database Catalog created automatically when an environment is activated, all compaction takes place on Cloudera Base on premises.

To change this interval:

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. Go to Database Catalog ⋮ Edit CONFIGURATIONS Metastore and select hive-site from the drop-down menu.
3. Search for the hive.compactor.check.interval KEY.
4. Add your preferred check interval in the associated VALUE field in seconds.
5. Click Apply Changes. The services are automatically updated with the new configuration.

Worker

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.

Cleaner

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.

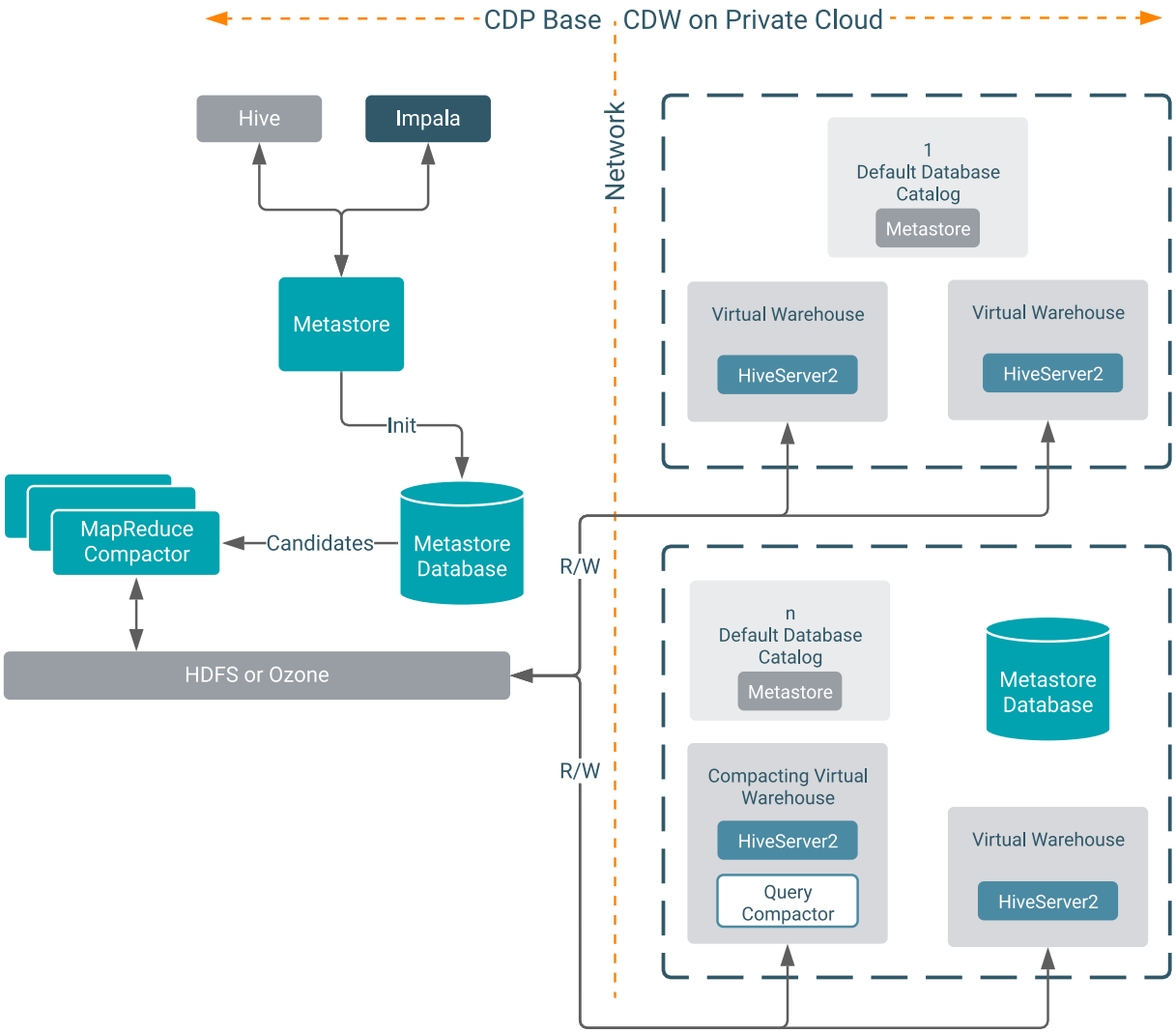
How compaction interacts with Cloudera Base on premises

In Cloudera Base on premises, the initiator and cleaner processes also run in the metastore as they do in Cloudera Data Warehouse on premises. However, the worker process runs in HiveServer2 as a MapReduce task so its progress can be viewed in YARN.

In Cloudera Data Warehouse, the initiator and cleaner processes run in the Database Catalog, which is the Cloudera Data Warehouse UI construct that equates to the metastore. The default Database Catalog, which is created by the system when you activate an environment in Cloudera Data Warehouse, maintains a connection with Cloudera Base on premises and all compaction jobs for the default Database Catalog run on Cloudera Base on premises. However, subsequent Database Catalogs that are created do not maintain a connection to Cloudera Base on premises and compaction runs entirely in Cloudera Data Warehouse. Also in Cloudera Data Warehouse, the worker process that performs the compaction work runs in HiveServer2, which equates to a Hive Virtual Warehouse. However, compaction performed by the worker process in Hive Virtual Warehouses consists of queries instead of MapReduce tasks.

Cloudera Data Warehouse on premises Compaction Architecture

This diagram illustrates how the components that perform compaction interact on Cloudera Data Warehouse on premises. In Cloudera Data Warehouse on premises, all compaction tasks for the default Database Catalog are performed on Cloudera Base on premises.



Considerations for using compaction on Cloudera Data Warehouse on premises

The first Hive Virtual Warehouse you create in Cloudera Data Warehouse on premises for a Database Catalog (not including the default Database Catalog) is automatically set as the compactor and performs all compaction work for subsequent Virtual Warehouses (Hive or Impala) created under that Database Catalog.

Consequently, 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:

- In the case of the default Database Catalog, all compaction takes place on Cloudera Base on premises so you do not need to consider compaction queries for the Virtual Warehouses that use the default Database Catalog.
- 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.

Changing compactor configuration for Hive Virtual Warehouses on Cloudera Data Warehouse on premises

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 premises. You can change which Hive warehouse performs compaction to load-balance this workload as necessary.

About this task

In Cloudera Data Warehouse on premises, data compaction is performed on HiveServer2, which equates to the Hive Virtual Warehouse construct in the UI. This means that compaction is essentially query execution. Compaction runs an INSERT statement created from the output of a SELECT statement and runs in the Hive Virtual Warehouse assigned as the compactor, thereby re-writing the data. The Hive Virtual Warehouse, configured as the compactor, delivers the query capacity to perform this. Therefore, when you size the Hive Virtual Warehouse that performs compaction, you must take into consideration the extra workload to run the compaction queries. That extra workload needs to be considered in addition to your other query workloads on the Hive Virtual Warehouse that is configured as the compactor.

Important: All compaction tasks for the warehouses that use the default Database Catalog, which is the Database Catalog automatically created for you when you activate an environment for Cloudera Data Warehouse, are performed on Cloudera Base on premises and do not affect the performance of Virtual Warehouses that use the default Database Catalog. For all other Database Catalogs that you create, you must consider the compaction query workload for the Hive Virtual Warehouse that performs compaction tasks.

Before you begin

One of the Hive Virtual Warehouses must be configured as the compactor for the associated Database Catalog (excluding the default Database Catalog whose commpaction is performed on Cloudera Base on premises). 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 configured as the compactor Virtual Warehouse for a Database Catalog. Compaction tasks must 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 Data Warehouse service.
2. On the **Overview** page, select the Hive Virtual Warehouse that you want to set as the compactor, and click ⋮ .
3. In the options menu, select Set Compactor.

Related Information

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 the Cloudera Management Console 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.

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. Alternatively, you can also access Grafana from the Management Console.

Before you begin

You must have an activated Cloudera Data Warehouse environment to view dashboards for Cloudera Data Warehouse service resources in Grafana.

About this task

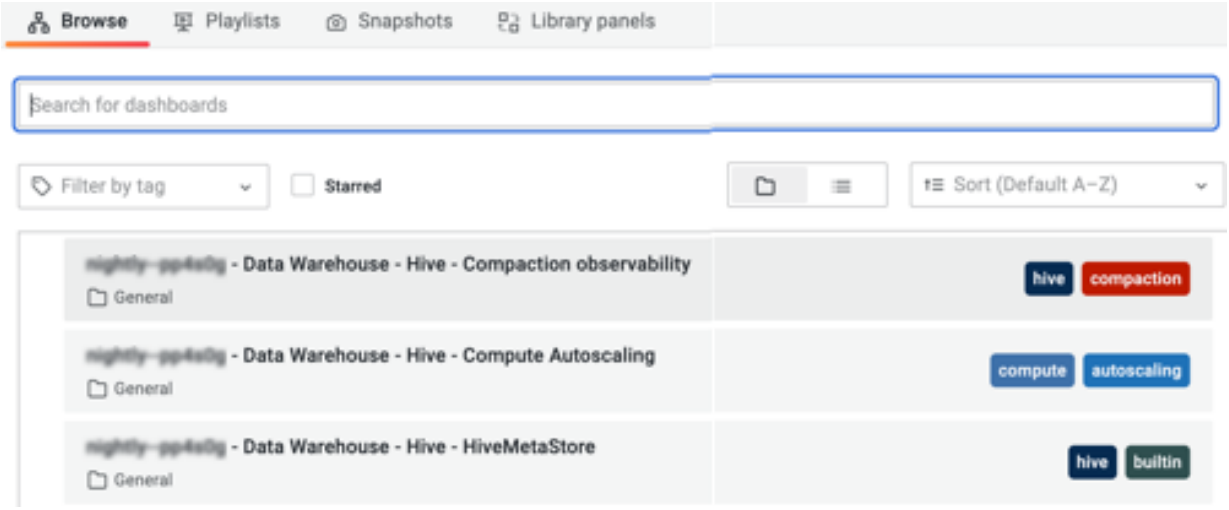
You can also access the Grafana dashboards from the Cloudera Management Console by going to the Dashboard page and clicking Monitoring Dashboard.

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 <Environment Name>- Data Warehouse - Hive - Compaction observability dashboard group to view all the dashboards related to Hive compactions.



Related Information

Configuring compaction alert rules

You can use the Cloudera Management Console to define and manage 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.

You can create custom alert rules based on your requirements. Further, you can edit, delete, enable, or disable an alert. Ensure that you have administrator privileges to configure alert rules.

For information about defining a new alert rule, see Configuring alert rules.

For information about managing alert rules, see Additional operations on the alert rules.

Related Information

Configuring alert rules

Additional operations on the alert rules

Configure alert receivers

As an 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 Hive Compaction Observability dashboard in Grafana. Learn how you can use the Cloudera Management Console for this configuration.

You can configure alert receivers in the Cloudera Management Console to trigger automated system-specific event notifications through external services such as emails, Slack channel messages, webhook notifications, PagerDuty messages, or SNMP traps. By configuring an alert receiver, you specify the details of an external service through which Cloudera Management Console forwards the notification to the specified destination.

For more information, see Configuring alert receivers using Cloudera Management Console.

Related Information
Configuring alert receivers

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

Note: The ability to rebuild a Virtual Warehouse is in Technical Preview and is not recommended for production deployments. Cloudera recommends that you try this feature in test or development environments.

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 2023.0.14.1 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.

Performance tuning for Trino Virtual Warehouse

This section provides guidance on performance tuning for Trino Virtual Warehouses, covering topics such as increasing preferred nodes for caching files, configuring resource management to address memory issues, and setting up spill memory properties for memory-intensive queries.

Note: Trino is in Technical Preview and is not ready for production deployments. Cloudera recommends trying this feature in test or development environments and encourages you to provide feedback on your experiences.

Increasing the number of preferred nodes for caching files

Query processing in Trino is distributed into different stages where tasks are processed by different nodes in the cluster. You can limit the number of hosts that are required to process these tasks with the fs.cache.preferred-hosts-count property.

If you are using the Hive or Iceberg connector and if the workload from these tables are high, then a high throughput is required, which can be achieved by increasing the value of the fs.cache.preferred-hosts-count connector property in the connector configuration details.

The value must be increased as concurrency increases. The default value of 2 may not be sufficient if you are running a high workload. For example, if you are running 40 nodes with a concurrency of 5, consider increasing the value of fs.cache.preferred-hosts-count to 20. You can increase this value up to a maximum limit of 40 (maximum worker nodes allowed for caching files).

Note that the value depends on the workload that you are trying to query through the connector and on the number of parallel queries that you expect to run simultaneously. Therefore, tune the parameter according to your workload needs.

Configuring resource management

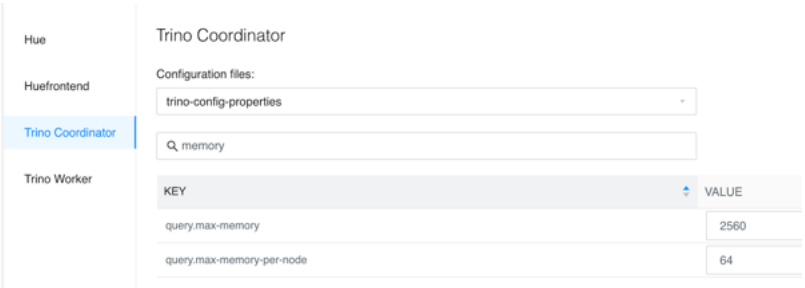
If you are experiencing memory issues and notice queries failing, it is recommended that you increase the allocated query memory for the Trino Coordinator and Worker.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the **Virtual Warehouses** tab, locate the Trino Virtual Warehouse and click ⋮ Edit .
   The **Trino Virtual Warehouse Details** page is displayed.
3. Click the **Configurations** tab, select Trino Coordinator, and select trino-config-properties from the Configuration files drop-down menu.
4. Search for the following properties and modify the values as recommended:
   - query.max-memory — Maximum amount of user memory that can be used by a query across the entire cluster.

     Recommended value: query.max-memory-per-node * number of worker nodes. For example, for 40 nodes, the value is 2560 GB.
   - query.max-memory-per-node — Maximum amount of user memory that can be used by a query on a worker node.

     Recommended value: 64 GB



5. Click Apply Changes.
6. From the **Configuration** tab, click Trino Worker and then select trino-config-properties from the **Configuration files** drop-down menu.
7. Search for the same memory related properties that are specified in Step 4 and set the appropriate values.

8. Click Apply Changes.

Configuring spill memory

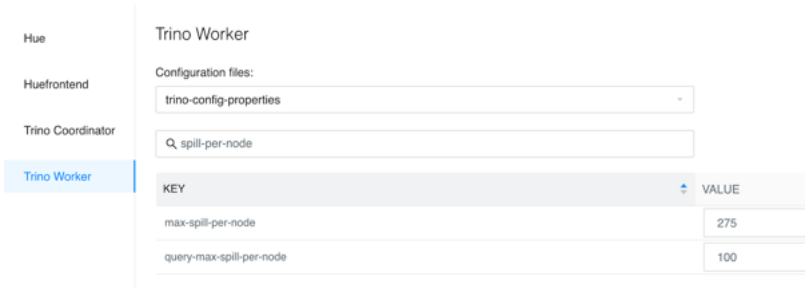
During memory intensive queries, if you notice issues related to the spill memory, you can configure certain properties to facilitate running of queries that require memory exceeding per query or per node limits.

Procedure

1. Log in to the Cloudera Data Warehouse service as DWAdmin.
2. Go to the **Virtual Warehouses** tab, locate the Trino Virtual Warehouse and click ⋮ Edit .
   The **Trino Virtual Warehouse Details** page is displayed.
3. Click the **Configurations** tab, select Trino Worker, and select trino-config-properties from the Configuration files drop-down menu.
4. Click Add Custom Configuration and add the following properties along with the recommended values:
   - query-max-spill-per-node — Maximum spill space that can be used by a single query on a single worker node.

     Recommended value: 100 GB
   - max-spill-per-node — Maximum spill space that can be used by all queries on a single worker node.

     Recommended value: 275 GB



Note: The spill space is shared with the Filesystem cache (fs-cache).

5. Click Apply Changes.