Cloudera Data Warehouse Private Cloud 1.4.1

CDW Auto-scaling

Date published: 2020-08-17
Date modified: 2022-11-18



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2025. All rights reserved.

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

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

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

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

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

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

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

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

Contents

Hive query isolation ETL jobs....4

Auto-scaling in private cloud environments....6

Hive auto-scaling on private clouds....6

Impala auto-scaling on private clouds....9

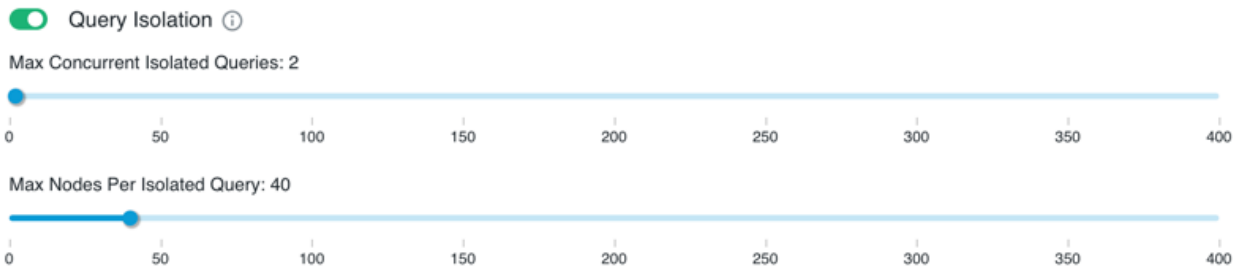
Hive query isolation ETL jobs

You can configure the SQL engine to isolate queries that involve lengthy scanning. Isolating these queries dedicates compute resources to handle these scans, leaving other resources free to handle less intensive jobs, such as ad-hoc queries.

If you enable Query Isolation in a Hive Virtual Warehouse, any query that exceeds the hive.query.isolation.scan.size.threshold value runs in isolation. An isolated standalone executor group is spawned to run the data-intensive query. Concurrency auto-scaling, auto-suspending, and auto-resume features of the core executor groups are not impacted by any isolated queries that are running.

Enabling query isolation

While creating or editing a Hive Virtual Warehouse, you can enable or disable query isolation and configure parameters, such as the maximum concurrent isolated queries and maximum nodes per isolated query. These parameters allow you to control resource usage and costs.



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

Specifies the maximum number of queries that can spawn dedicated executor groups at one time. 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)

Specifies the maximum number of executors that can be created for each data-intensive query.

Determining the default scan size threshold

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 4TB 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.

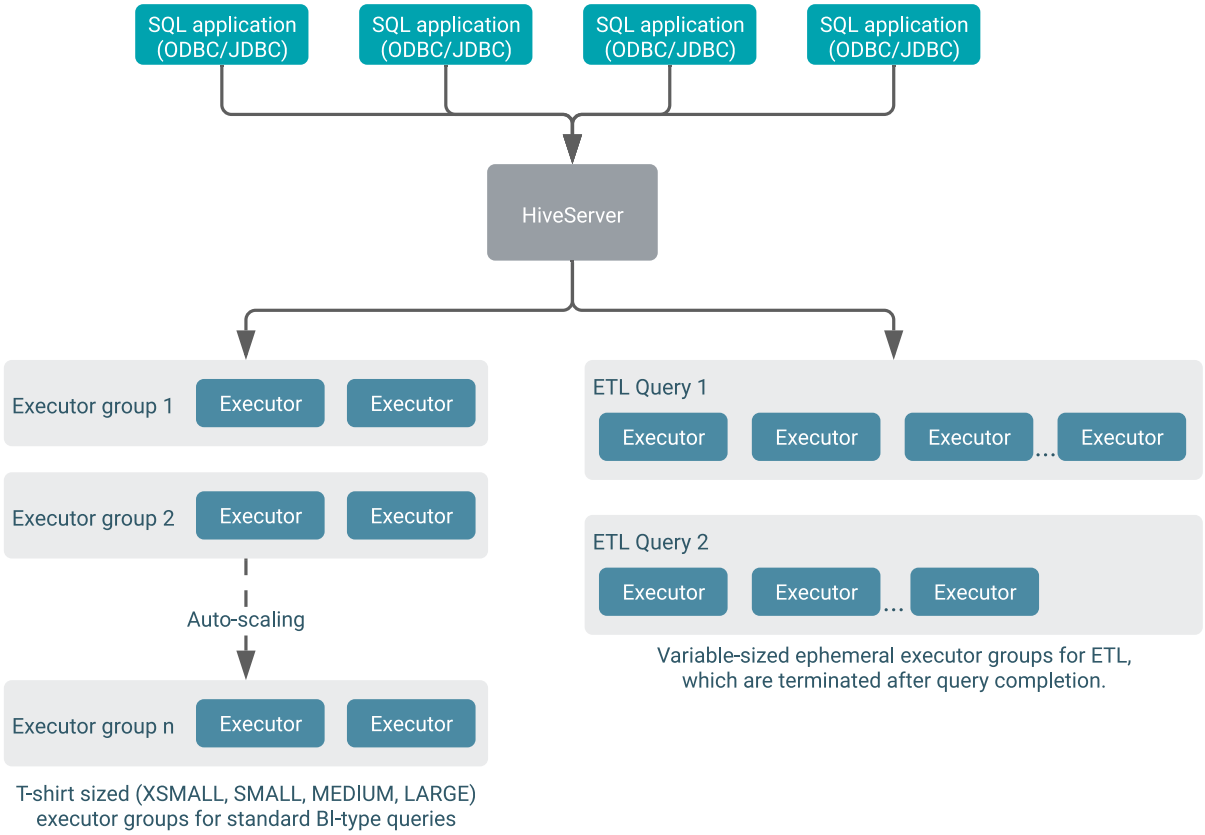
Assuming the original data-intensive "scan-heavy" isolated query is still running, if HiveServer receives an additional "scan-heavy" query, which scans more than 4TB of data, another isolated executor group is spawned to run the additional query. However, if two additional "scan-heavy" queries are received, 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 one of the other isolated queries has finished executing.

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

Query isolation process

After you have specified 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 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 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.

Related Information

Auto-scale threshold settings
Tuning Hive Virtual Warehouses
Virtual Warehouse sizing requirements

Virtual Warehouse IP address and cloud resource requirements

Auto-scaling in private cloud environments

This section describes how Virtual Warehouse auto-scaling works in Cloudera Data Warehouse (CDW) Private Cloud.

Auto-scaling enables both scaling up and scaling down of Virtual Warehouse instances so they can meet your varying workload demands and free up resources on the OpenShift cluster for use by other workloads. The following topics describe how auto-scaling works in Hive and Impala Virtual Warehouses in CDW Private Cloud.

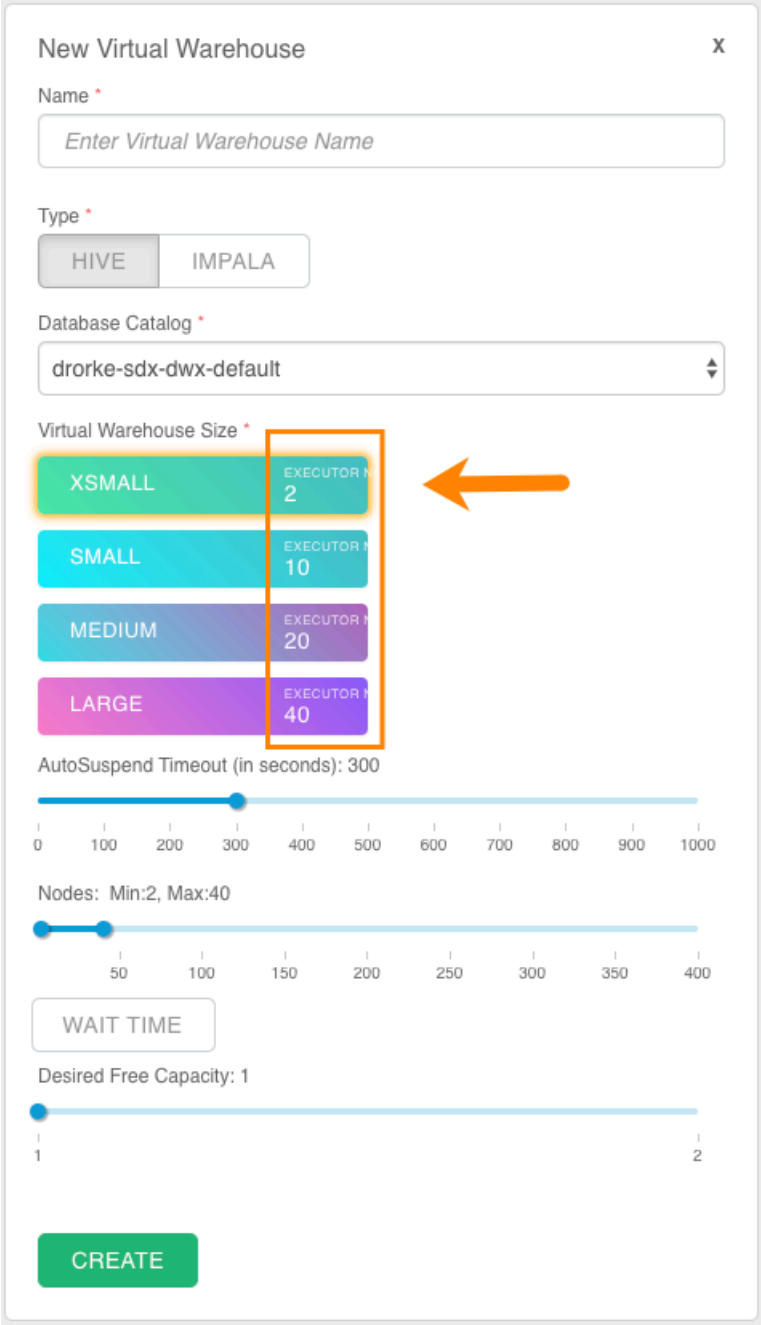
Hive auto-scaling on private clouds

This topic describes how auto-scaling works for Hive Virtual Warehouses in Cloudera Data Warehouse (CDW) Private Cloud.

Auto-scaling for Hive Virtual Warehouses in CDW Private Cloud service 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 for distribution to executor nodes.
- 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 not a one-to-one correspondence. 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.

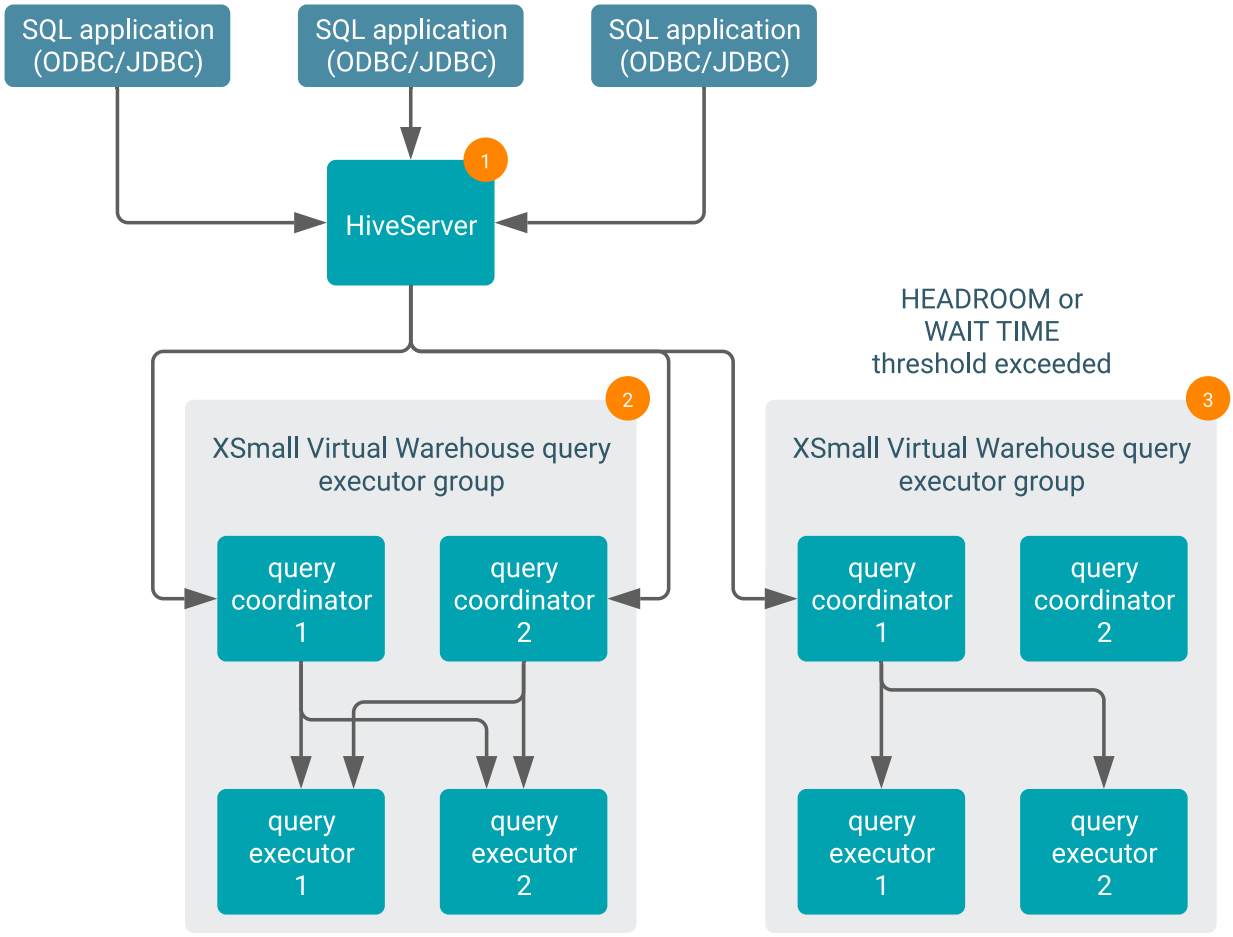
- Query executor containers: Can run multiple tasks of one or multiple different queries. Query executor nodes determine the throughput of your system. They are the executors which are the unit of sizing for Virtual Warehouses:



- 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.

Auto-scaling process

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, Data Analytics Studio (DAS), 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 tries to optimize for cache locality.
3. Depending on whether WAIT TIME has been set to manage auto-scaling, additional query executor groups are added when the auto-scaling threshold has been exceeded.

   WAIT TIME sets how long queries wait in the queue to run. A query is queued if it arrives on HiveServer and no coordinator is available. For example, if WaitTime Seconds is set to 10, queries are waiting to run in the queue for 10 seconds, the warehouse auto-scales up and an additional executor group is deployed until the maximum setting for Nodes:Min:, Max: has been reached. This ensures that your Virtual Warehouse does not take resources away from other workloads.

When no queries are being sent to an executor group, it scales down and the nodes are released. When all executor groups are scaled back to the original executor group created when the warehouse was created and its coordinators and executors are idle, the Virtual Warehouse is suspended according to the value set for AutoSuspend Timeout.

AutoSuspend Timeout

You set the AutoSuspend Timeout when you create a Virtual Warehouse:



This 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. AutoSuspend Timeout is independent of the auto-scaling process and only applies to the original executor group and not to any additional executor groups that are created as a result of auto-scaling.

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 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 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 will not occur until cluster resources are freed up.

Impala auto-scaling on private clouds

Your Impala Virtual Warehouse in Cloudera Data Warehouse (CDW) 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.

Impala Virtual Warehouses use the following processes to support auto-scaling:

- 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 likely 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. When you enable legacy multithreading , the Virtual Warehouse can run 12 queries per executor group. 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 would 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 would still contain the 20 executors required to run an average query, but you would have two executor groups to handle the

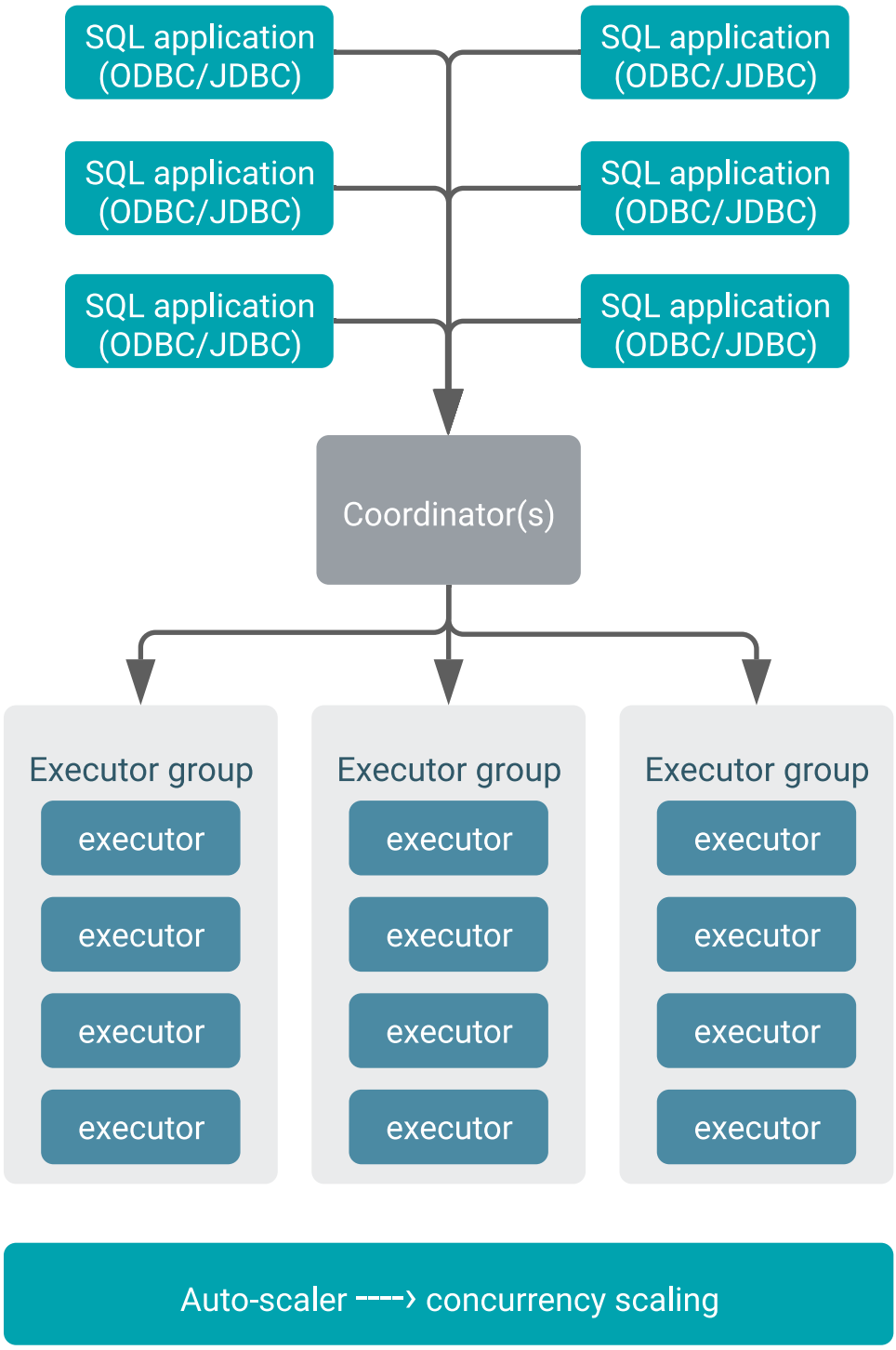
volume of queries in your workloads. If the total number of executor nodes auto-scales up to 80, then there will be 4 executor groups containing 20 executor nodes each.

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.

Auto-scaling process

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 for 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: turn on Disable Auto Suspend on the Virtual Warehouse:



The time CDW 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.