Cloudera Data Warehouse Public Cloud

CDW Auto-scaling

Date published: 2024-01-01
Date modified: 2024-01-01

https://docs.cloudera.com/

Legal Notice

Contents

Virtual Warehouse auto-scaling

The way Hive Virtual Warehouse performs auto-scaling is different from the Impala Virtual Warehouse. Learning how the auto-scaling works helps you choose the type of Virtual Warehouse and correctly configure the auto-scaling to handle your jobs.

Key features describes auto-scaling in Cloudera and its benefits. You need to know specifics about how auto-scaling works in the type of Virtual Warehouse you choose. The number of queries running concurrently and the query complexity determine how Hive Virtual Warehouse auto-scaling works. A BI-type query does not require the same resources and scaling process as an ETL-type query. A complex ETL-type query requires intensive data scanning. You can configure the auto-scaling method used by the Hive Virtual Warehouse based on the number of concurrently running queries or the size of data scanned for a query. There are two methods of Hive Virtual Warehouse autoscaling:

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

Impala Virtual Warehouse auto-scaling is designed for concurrently running BI queries to make resources available for queued queries. Like Hive auto-scaling methods, coordinators and executors control Impala auto-scaling.

Workload Aware Auto-Scaling (WAAS) is available for Impala Virtual Warehouses as a technical preview. Previewing this feature requires an entitlement. Contact your account team if you are interested in previewing the feature.

Understanding executors

In AWS environments, an executor is an Elastic Compute Cloud (EC2) r5d.4xlarge instance for Hive and Impala compute nodes. For more information, see "AWS resources created for Data Warehouse". In Azure environments, an executor is an Azure E16 v3 instance. For more information, see "Virtual Warehouse sizing requirements for public cloud environments".

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 you configured during creation. For example, if you created the Virtual Warehouse and configured the MEDIUM size, which has 20 executor nodes, then each executor group contains 20 executors.

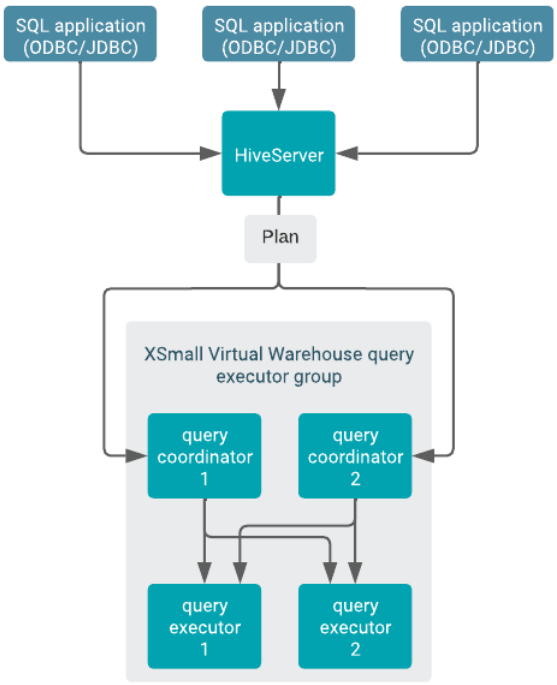
When you stop running queries, resources scale down, and executors are released. When all executor groups are scaled back, executors are idle, and after a configured period of idle time (AutoSuspend Timeout) elapses, the Virtual Warehouse is suspended. If a query is executed on a suspended Virtual Warehouse, 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.

Hive VW concurrency auto-scaling

The Hive and Impala Virtual Warehouses use concurrency auto-scaling under certain conditions, but the auto-scaling implementation is different. You learn about the Hive Virtual Warehouse concurrency auto-scaling, which helps you better configure a Hive Virtual Warehouse for running BI queries.

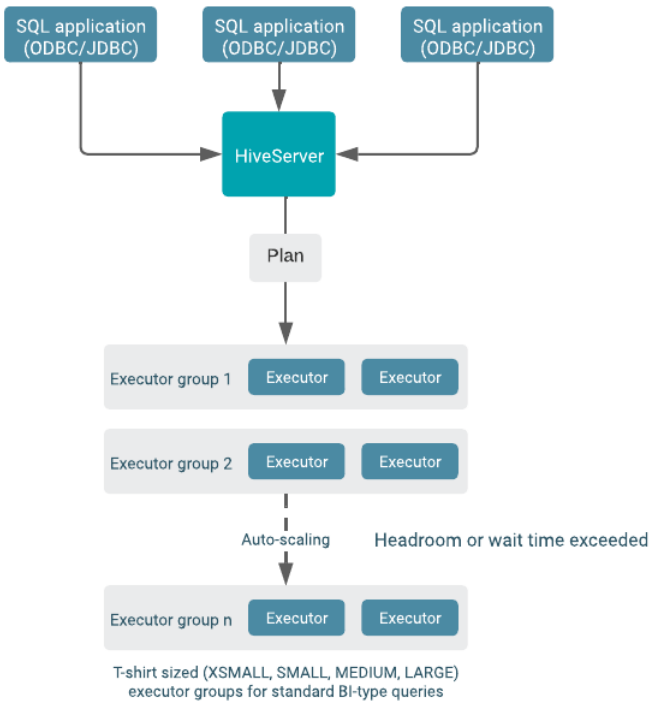
Concurrency auto-scaling is designed for running BI queries. When the query load increases, auto-scaling increases. The query load grows proportionally with the number of concurrent queries and query complexity. When you run the BI query, HiveServer generates a preliminary query execution plan.

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, as shown in the following diagram of an XSmall Virtual Warehouse, for example:



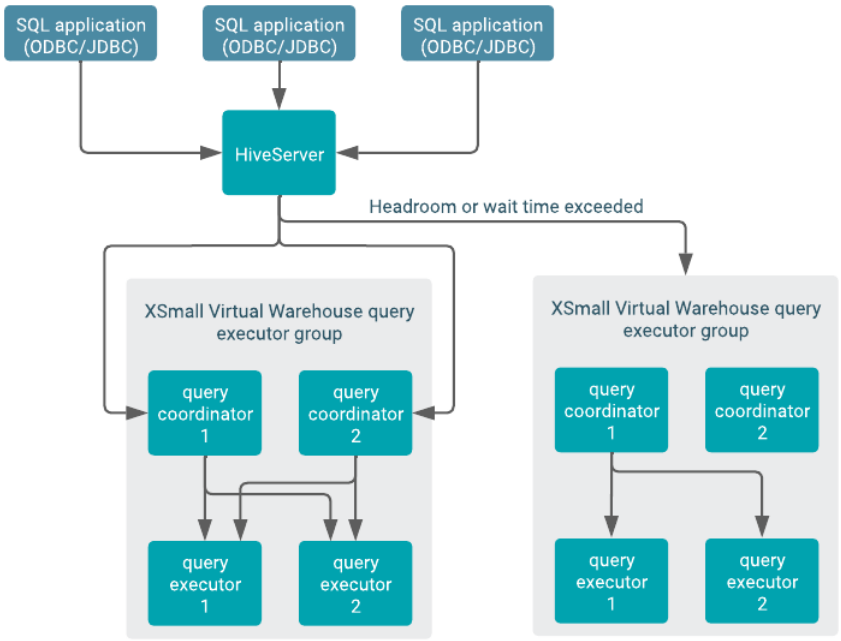
A single query on an idle Virtual Warehouse does not cause auto-scaling. The query uses the resources available at the time of the query submission. Only subsequent queries can cause auto-scaling.

Auto-scaling occurs when either the headroom, or wait time, exceeds the threshold you configure. Additional query executor groups are added to do the scaling, as shown in the following diagram.



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.

The following diagram shows the same basic auto-scaling scenario as the diagram above but from a different perspective. In this diagram the query coordinator perspective, instead of the executor group, is shown.



There are as many query coordinators as executors, but there is not a one-to-one correspondence. 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.

Related Information

Tuning Hive LLAP Virtual Warehouses

Virtual Warehouse sizing requirements

Virtual Warehouse IP address and cloud resource requirements

Hive VW isolation auto-scaling

The Hive Virtual Warehouse in Cloudera Data Warehouse (CDW) Public Cloud bases its auto-scaling on the size of data scanned for a query. Queries that scan amounts of data that exceed the hive.query.isolation.scan.size.threshold setting use the query isolation method of auto-scaling.

To run ETL-type queries, you turn on query isolation when you configure the Hive Virtual Warehouse. Scaling occurs as follows:

- If the query does not exceed the hive.query.isolation.scan.size.threshold, concurrency scaling occurs.
- If the query does exceed the hive.query.isolation.scan.size.threshold, query isolation scaling occurs.

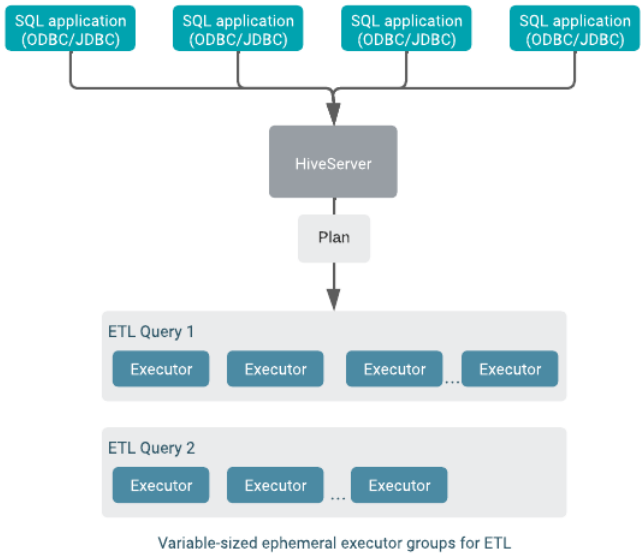
HiveServer generates a preliminary query execution plan.

The Hive Virtual Warehouse adds a new dedicated executor group with the right number of executors to handle the query instead of using the number associated with the T-shirt size you configured. This isolated executor group is used only under the following conditions:

- You run a single ETL-type query.
- The query exceeds the hive.query.isolation.scan.size.threshold.

Otherwise, the concurrency method of auto-scaling is used.

The following diagram shows the isolation method of auto-scaling:



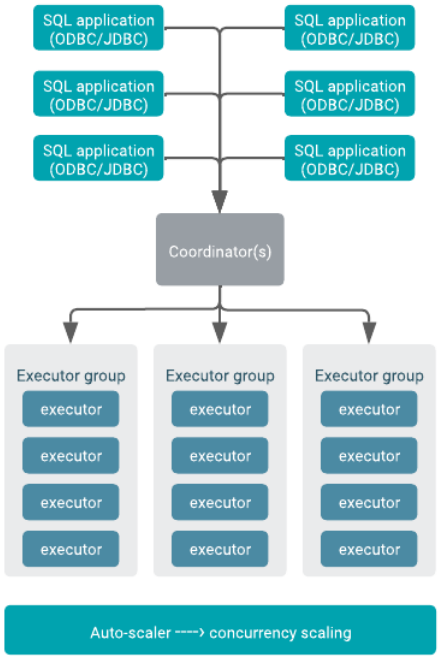
Variable-sized ephemeral executor groups for ETL

The variable-sized, ephemeral executor groups for ETL terminate after the query completes.

Impala VW auto-scaling

The Impala Virtual Warehouse, like the Hive Virtual Warehouse does concurrency auto-scaling, but in a different way. Learning how the auto-scaling works helps you choose the type of Virtual Warehouse and correctly configure the auto-scaling to handle your jobs.

The Impala Virtual Warehouse auto-scaling works with coordinators and executors to make resources available for queued queries, as shown in the following diagram:



An auto-scaler process (a Kubernetes pod), shown at the bottom of the diagram, monitors Impala running in the Virtual Warehouse to determine the necessary resources. When the auto-scaler detects an imbalance in resources, it sends a request to Kubernetes to increase or decrease the number of executor groups in the Virtual Warehouse.

Executor groups execute queries. By default Impala Virtual Warehouses can run 3 large queries per executor group. For most read-only queries the default setting of 3 queries per executor group is sufficient. Executors can handle more queries that are simpler and that do not utilize concurrency on the executor.

Executor processes execute query fragments. Executor processes run on executor nodes, the unit of sizing for Virtual Warehouses. Each executor node runs one executor.

Related Information

Tuning Impala Virtual Warehouses

Virtual Warehouse sizing requirements

Virtual Warehouse IP address and cloud resource requirements

Workload Aware Auto-Scaling in Impala (Preview)

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

The Workload Aware Auto-Scaling (WAAS) technical preview improves auto-scaling for the following types of workloads:

- Mixed or unknown workloads
- On-premises workloads you are migrating to CDW
- 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.

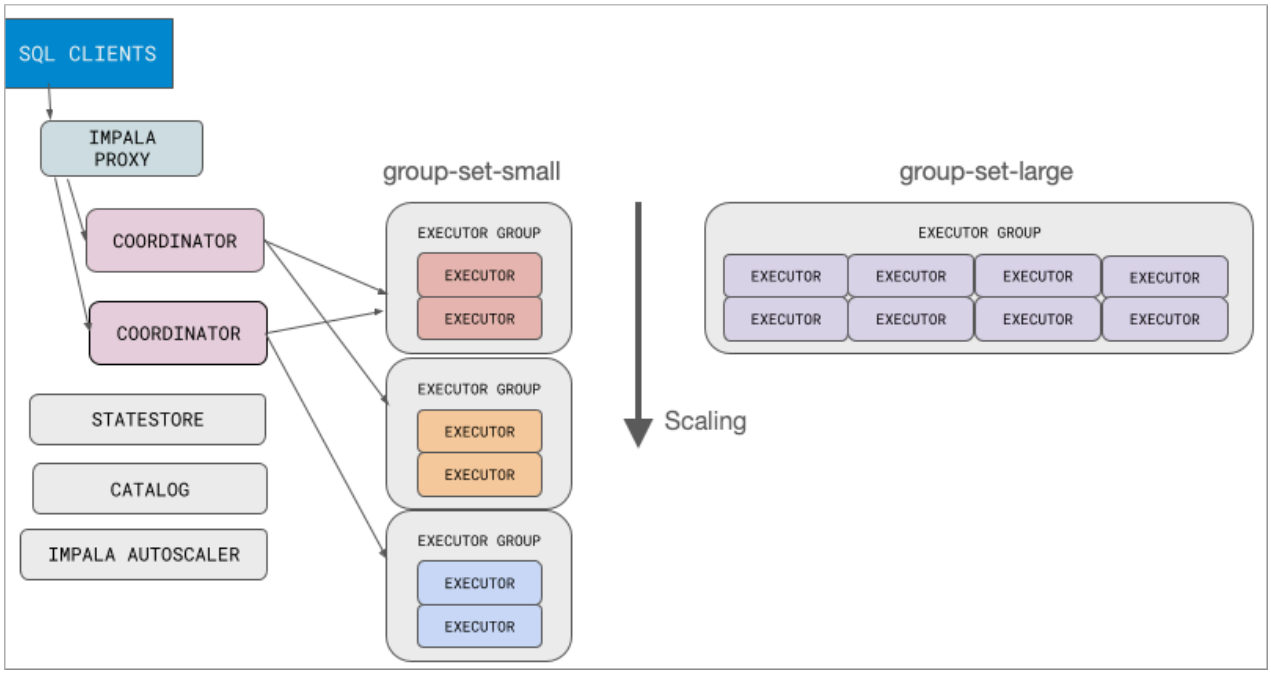
With WAAS, if you have mixed workloads, you define constraints for the overall warehouse size. You can add multiple sizes of executor group sets to spin up if needed by the workload. Within the executor group set, executor groups spin up or shut down as warranted by the workload. Each executor group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

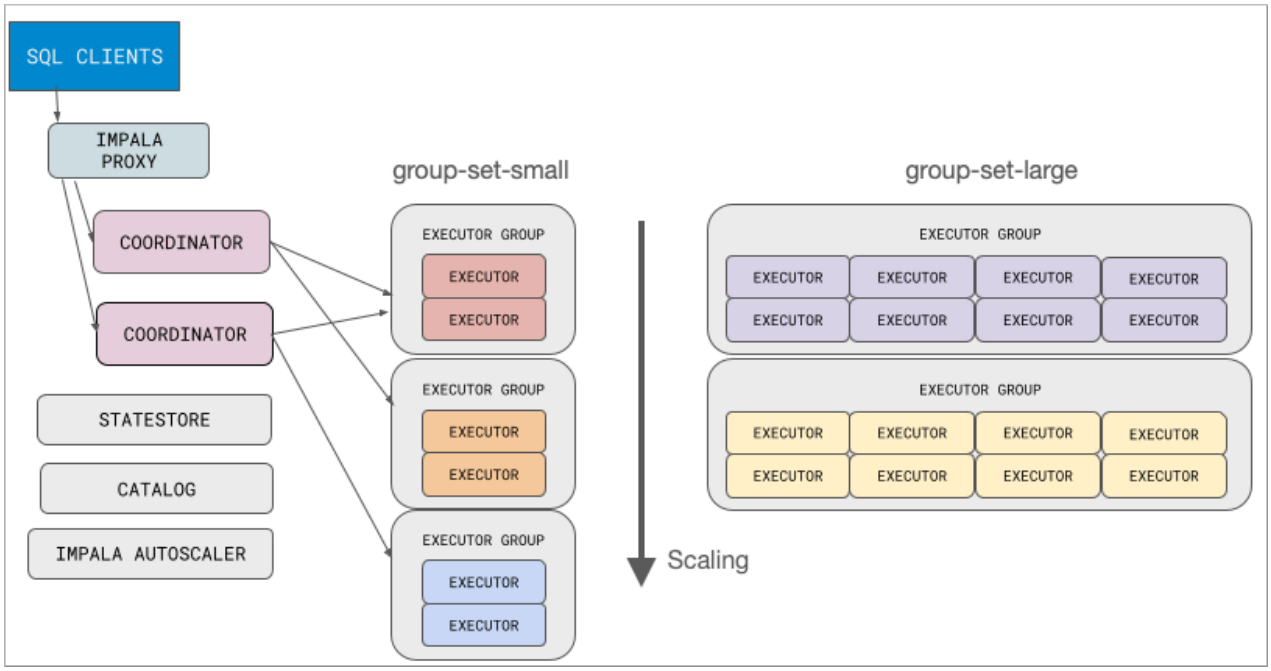
The graphic below depicts two executor groups sets in a WAAS Virtual Warehouse:

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

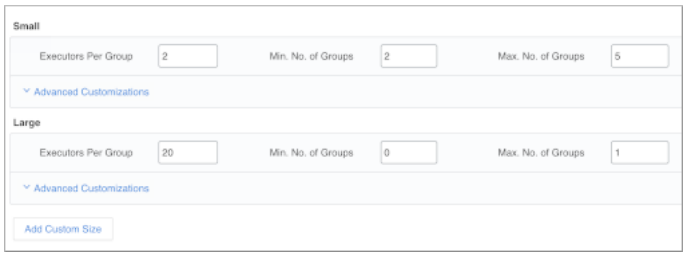
Within any executor group set are executor groups, all of the same size.



The following graphic depicts scaling that occurs when you run a large query and more capacity is needed. 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.

Table 1:

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

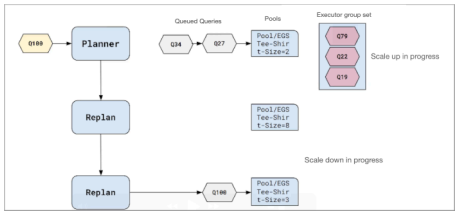
Workload Aware Auto-Scaling uses replanning, described in the next topic, to select an executor group set.

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 the following documentation:

- Data Hub: Dynamic Resource Pool Settings
- CDW: Configuring Admission Control and Configuring Resource Pools and Admission Control

Workload-aware autoscaling 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 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

Workload Aware Auto-Scaling 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.