Managing and Allocating Cluster Resources using Capacity Scheduler

Date published:
Date modified:



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. 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

Resource scheduling and management....6
YARN resource allocation of multiple resource-types....6
Hierarchical queue characteristics....6
Scheduling among queues....7
Application reservations....7
Resource distribution workflow....7
Resource allocation overview....9
Use CPU scheduling....10
Configure CPU scheduling and isolation....10
Use CPU scheduling with distributed shell....11
Use GPU scheduling....11
Configure GPU scheduling and isolation....11
Use GPU scheduling with distributed shell....12
Use FPGA scheduling....13
Configure FPGA scheduling and isolation....13
Use FPGA with distributed shell....14
Limit CPU usage with Cgroups....14
Use Cgroups....15
Enable Cgroups....15

Managing YARN Queue Manager....17
Configuring YARN Queue Manager dependency....17
Updating YARN Queue Manager Database Password....18
Accessing the YARN Queue Manager UI....19
Providing read-only access to Queue Manager UI....20
Configuring the embedded Jetty Server in Queue Manager....23

Managing queues....23
Adding queues using YARN Queue Manager UI....24
Configuring cluster capacity with queues....29
Configuring the resource capacity of root queue in absolute mode....31
Mixed resource allocation mode (Technical Preview)....33
Setting capacity using mixed resource allocation mode (Technical Preview)....33
Changing resource allocation mode....34
Starting and stopping queues....35
Deleting queues....36
Setting queue priorities....36

Configuring scheduler properties at the global level....37
Setting global maximum application priority....39
Configuring preemption....39
Enabling Intra-Queue preemption....41
Enabling LazyPreemption....41
Setting global application limits....42
Setting default Application Master resource limit....42
Enabling asynchronous scheduler....43

Configuring queue mapping to use the user name from the application tag using Cloudera Manager.....43
Configuring NodeManager heartbeat.....44
Configuring data locality.....45
Setting Maximum Parallel Application.....45
Setting maximum parallel application limits.....47

Configuring per queue properties.....47
Setting user limits within a queue.....50
Setting Maximum Application limit for a specific queue.....52
Setting Application-Master resource-limit for a specific queue.....53
Setting maximum parallel application limits for a specific queue.....53
Controlling access to queues using ACLs.....53
Enabling preemption for a specific queue.....54
Enabling Intra-Queue Preemption for a specific queue.....55
Setting ordering policies within a specific queue.....55
Configure queue ordering policies.....56

Autoscaling clusters.....57
Autoscaling behavior.....58
Configuring autoscaling.....59

Dynamic Queue Scheduling.....63
Creating a new Dynamic Configuration.....63
Managing Dynamic Configurations.....67
How to read the Configurations table.....68
Handling Dynamic Configuration conflicts.....69
Revalidating Dynamic Configurations.....69
Dynamic Configurations execution log.....70

Managing placement rules.....71
Placement rule policies.....71
How to read the Placement Rules table.....74
Creating placement rules.....75
Example - Placement rules creation.....78
Reordering placement rules.....79
Editing placement rules.....79
Deleting placement rules.....80
Enabling override of default queue mappings.....80

Managing dynamic queues.....81
Managed Parent Queues.....82
Converting a queue to a Managed Parent Queue.....82
Enabling dynamic child creation in weight mode.....83
Disabling dynamic child creation in weight mode.....85
Managing dynamic child creation enabled parent queues.....85
Managing dynamically created child queues.....86
Deleting dynamically created child queues.....86
Disabling auto queue deletion globally.....87
Disabling queue auto removal on a queue level.....87
Configuring the queue auto removal expiration time.....88
Deleting dynamically created child queues manually.....88

Partition configuration.....88
Enabling node labels on a cluster to configure partition.....90
Creating partitions.....91
Assigning or unassigning a node to a partition.....92
Viewing partitions.....92
Associating partitions with queues.....93
Disassociating partitions from queues.....96
Deleting partitions.....97
Setting a default partition expression.....100
Using partitions when submitting a job.....100

Resource scheduling and management

You can manage resources for the applications running on your cluster by allocating resources through scheduling, limiting CPU usage by configuring cgroups, and partitioning the cluster into subclusters using partitions, and launching applications on Docker containers.

The *CapacityScheduler* is responsible for scheduling. The *CapacityScheduler* is used to run Hadoop applications as a shared, multi-tenant cluster in an operator-friendly manner while maximizing the throughput and the utilization of the cluster.

The *ResourceCalculator* is part of the YARN *CapacityScheduler*. If you have only one type of resource, typically a CPU virtual core (vcore), use the DefaultResourceCalculator. If you have multiple resource types, use the Dominant ResourceCalculator.

YARN resource allocation of multiple resource-types

You can manage your cluster capacity using the Capacity Scheduler in YARN. You can use the Capacity Scheduler's DefaultResourceCalculator or the DominantResourceCalculator to allocate available resources.

The fundamental unit of scheduling in YARN is the queue. The capacity of each queue specifies the percentage of cluster resources available for applications submitted to the queue. You can set up queues in a hierarchy that reflects the database structure, resource requirements, and access restrictions required by the organizations, groups, and individuals who use the cluster resources.

You can use the default resource calculator when you want the resource calculator to consider only the available memory for resource calculation. When you use the default resource calculator (DefaultResourceCalculator), resources are allocated based on the available memory.

If you have multiple resource types, use the dominant resource calculator (DominantResourceCalculator) to enable CPU, , and memory scheduling. The dominant resource calculator is based on the Dominant Resource Fairness (DRF) model of resource allocation. DRF is designed to fairly distribute CPU, and memory resources among different types of processes in a mixed-workload cluster.

For example, if User A runs CPU-heavy tasks and User B runs memory-heavy tasks, the DRF allocates more CPU and less memory to the tasks run by User A, and allocates less CPU and more memory to the tasks run by User B. In a single resource case, in which all jobs are requesting the same resources, the DRF reduces to max-min fairness for that resource.

Hierarchical queue characteristics

You must consider the various characteristics of the Capacity Scheduler hierarchical queues before setting them up.

There are two types of queues: parent queues and leaf queues.

- Parent queues enable the management of resources across organizations and sub- organizations. They can contain more parent queues or leaf queues. They do not themselves accept any application submissions directly.
- Leaf queues are the queues that live under a parent queue and accept applications. Leaf queues do not have any child queues, and therefore do not have any configuration property that ends with ".queues".
- There is a top-level parent root queue that does not belong to any organization, but instead represents the cluster itself.
- Using parent and leaf queues, administrators can specify capacity allocations for various organizations and sub-organizations.

Scheduling among queues

Hierarchical queues ensure that guaranteed resources are first shared among the sub-queues of an organization before any remaining free resources are shared with queues belonging to other organizations. This enables each organization to have control over the utilization of its guaranteed resources.

- At each level in the hierarchy, every parent queue keeps the list of its child queues in a sorted manner based on demand. The sorting of the queues is determined by the currently used fraction of each queue’s capacity (or the full-path queue names if the reserved capacity of any two queues is equal).
- The root queue understands how the cluster capacity needs to be distributed among the first level of parent queues and invokes scheduling on each of its child queues.
- Every parent queue applies its capacity constraints to all of its child queues.
- Leaf queues hold the list of active applications (potentially from multiple users) and schedules resources in a FIFO (First In, First Out) manner, while at the same time adhering to capacity limits specified for individual users.

Application reservations

For a resource-intensive application, the Capacity Scheduler creates a reservation on a cluster node if the node's free capacity can meet the particular application's requirements. This ensures that the resources are utilized only by that particular application until the application reservation is fulfilled.

The Capacity Scheduler is responsible for matching free resources in the cluster with the resource requirements of an application. Many times, a scheduling cycle occurs such that even though there are free resources on a node, they are not sized large enough to satisfy the application waiting for a resource at the head of the queue. This typically happens with high-memory applications whose resource demand for Containers is much larger than the typical application running in the cluster. This mismatch can lead to starving these resource-intensive applications.

The Capacity Scheduler reservations feature addresses this issue as follows:

- When a node reports in with a finished Container, the Capacity Scheduler selects an appropriate queue to utilize the newly available resources based on capacity and maximum capacity settings.
- Within that selected queue, the Capacity Scheduler looks at the applications in a FIFO order along with the user limits. Once a needy application is found, the Capacity Scheduler tries to see if the requirements of that application can be met by the node’s free capacity.
- If there is a size mismatch, the Capacity Scheduler immediately creates a reservation on the node for the application’s required Container.
- Once a reservation is made for an application on a node, those resources are not used by the Capacity Scheduler for any other queue, application, or Container until the application reservation is fulfilled.
- The node on which a reservation is made reports back when enough Containers finish running such that the total free capacity on the node now matches the reservation size. When that happens, the Capacity Scheduler marks the reservation as fulfilled, removes it, and allocates a Container on the node.
- In some cases another node fulfills the resources required by the application, so the application no longer needs the reserved capacity on the first node. In this situation, the reservation is simply cancelled.

To prevent the number of reservations from growing in an unbounded manner, and to avoid any potential scheduling deadlocks, the Capacity Scheduler maintains only one active reservation at a time on each node.

Resource distribution workflow

During scheduling, queues at any level in the hierarchy are sorted in the order of their current used capacity, and the available resources are distributed among them starting with queues that are currently the most under-served.

With respect to capacities alone, the resource scheduling has the following workflow:

- The more under-served a queue is, the higher the priority it receives during resource allocation. The most under-served queue is the queue with the least ratio of used capacity as compared to the total cluster capacity.

    - The used capacity of any parent queue is defined as the aggregate sum of used capacity of all of its descendant queues, recursively.
    - The used capacity of a leaf queue is the amount of resources used by the allocated Containers of all of the applications running in that queue.
- Once it is decided to give a parent queue the currently available free resources, further scheduling is done recursively to determine which child queue gets to use the resources -- based on the previously described concept of used capacities.
- Further scheduling happens inside each leaf queue to allocate resources to applications in a FIFO order.
    - This is also dependent on locality, user level limits, and application limits.
    - Once an application within a leaf queue is chosen, scheduling also happens within the application. Applications may have different priorities of resource requests.
- To ensure elasticity, capacity that is configured but not utilized by any queue due to lack of demand is automatically assigned to the queues that are in need of resources.

Resource Distribution Process in Relative Mode - Example

To understand the resource distribution workflow, consider the example of a 100-node cluster, each with 10 GB of memory allocated for YARN containers, for a total cluster capacity of 1000 GB (1 TB).

For example, in the Relative allocation mode, the "engineering" organization is assigned 60% of the cluster capacity, that is, an absolute capacity of 600 GB. Similarly, the "support" organization is assigned 100 GB, and the "marketing" organization gets 300 GB.

Under the "engineering" organization, capacity is distributed between the "development" team and the "qa" team in a 1:4 ratio. So "development" gets 120 GB, and 480 GB is assigned to "qa".

Now consider the following timeline of events:

- Initially, the entire "engineering" queue is free with no applications running, while the "support" and "marketing" queues are utilizing their full capacities.
- Users Sid and Hitesh first submit applications to the "development" leaf queue. Their applications are elastic and can run with either all of the resources available in the cluster, or with a subset of cluster resources (depending upon the state of the resource-usage).
    - Even though the "development" queue is allocated 120 GB, Sid and Hitesh are each allowed to occupy 120 GB, for a total of 240 GB.
    - This can happen despite the fact that the "development" queue is configured to be run with a capacity of 120 GB. Capacity Scheduler allows elastic sharing of cluster resources for better utilization of available cluster resources. Since there are no other users in the "engineering" queue, Sid and Hitesh are allowed to use the available free resources.
- Next, users Jian, Zhijie and Xuan submit more applications to the "development" leaf queue. Even though each is restricted to 120 GB, the overall used capacity in the queue becomes 600 GB -- essentially taking over all of the resources allocated to the "qa" leaf queue.
- User Gupta now submits an application to the "qa" queue. With no free resources available in the cluster, the user's application must wait.
    - Given that the "development" queue is utilizing all of the available cluster resources, Gupta may or may not be able to immediately get back the guaranteed capacity of his "qa" queue -- depending upon whether or not preemption is enabled.
- As the applications of Sid, Hitesh, Jian, Zhijie, and Xuan finish running and resources become available, the newly available Containers will be allocated to Gupta’s application.

This will continue until the cluster stabilizes at the intended 1:4 resource usage ratio for the "development" and "qa" queues.

From this example, you can see that it is possible for abusive users to submit applications continuously, and thereby lock out other queues from resource allocation until Containers finish running or get preempted. To avoid this scenario, Capacity Scheduler supports limits on the elastic growth of any queue. For example, you can restrict

the "development" queue from monopolizing the "engineering" queue capacity by setting the maximum capacity property.

To set the maximum capacity property based on this example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the development queue and select Edit Child Queues option.
3. Enter 40 in the Configured Value field.
4. Click Save.

Once this is set, users of the "development" queue can still go beyond their capacity of 120 GB, but they will not be allocated any more than 40% of the "engineering" parent queue's capacity (that is, 40% of 600 GB = 240 GB).

The capacity and maximum-capacity properties can be used to control sharing and elasticity across the organizations and sub-organizations utilizing a YARN cluster. Administrators should balance these properties to avoid strict limits that result in a loss of utilization, and to avoid excessive cross-organization sharing.

Resource allocation overview

You can allocate resources to queues by using either the Absolute mode, in which the actual units of vCores and memory resources are allocated, the Relative mode, in which resources are allocated as a percentage of total resources, or in the Weight mode, in which resources are allocated as a fraction of total resources or in the Mixed mode, in which you specify the resources in a combination of these allocation modes.

- Relative mode: In the YARN Queue Manager UI, the capacity value is shown as a percentage. When editing queue resources, capacities are specified in percentages. The capacity allocated to a set of sibling queues must equal 100% of the parent.
- Absolute mode: In the YARN Queue Manager UI, the capacity value is shown in units. When editing queue resources, a separate tab is displayed for each resource type. The total allocated capacity must be less than or equal to the capacity of the parent.
- Weight mode: In the YARN Queue Manager UI, the capacity value is shown as a numerical value and is suffixed with “w”. For example, 5w. When editing queue resources, capacities are specified in weights or fractions of total resources. The resources are divided based on how the queue's weight relates to the sum of configured weights under the parent.

  For example, the values set are:

  - Min user percentage = 20%
  - User limit faction = 4
  - Max am resource limit = 50%
  - Configure capacity = 4W
  - Max capacity = 100%
  - Preemption is disabled

  Calculation of cluster queues & its Total weights = 100w + 100w + 40w + 25w + 1w + 1w + 1w + 1w = 200w + 65w + 4w = 269w = 100% root capacity.

  And, the root capacity is equal to 18261.7 GB. Hence, 1w = (18261.7/269) = 67.89 GB

  Therefore, maximum capacity = 67.89 GB * 4w = 271.56 GB

  Note: If you have an existing managed queue in Relative mode, then conversion to Weight mode is not allowed. You must delete the managed parent queue and then convert from Relative mode to Weight mode.
- Mixed mode (Technical Preview): In this mode, you can specify the resources in mixed types by updating the XML configuration usingCloudera Manager. You can specify the actual units of vcores and memory resources for each queue or specify the percentage of the total resources used by each queue or specify a weight for each queue.

You can use a combination of these allocation modes. The queues under one parent can also mix their modes. For more information, see *Setting capacity using mixed resource allocation mode*.

Important: This is a technical preview feature and is considered under development. Do not use this in your production systems. If you have feedback, contact Support by logging a case on our Cloudera Support Portal. Technical preview features are not guaranteed troubleshooting and fixes.

Related Information

Changing resource allocation mode
Setting capacity using mixed resource allocation mode (Technical Preview)
Cloudera Support Portal
Adding queues using YARN Queue Manager UI
Configuring cluster capacity with queues

Use CPU scheduling

Cgroups with CPU scheduling helps you effectively manage mixed workloads.

Note: You should use CPU scheduling only in a Linux environment, because there is no isolation mechanism (cgroups equivalent) for Windows.

MapReduce jobs only

If you primarily run MapReduce jobs on your cluster, enabling CPU scheduling does not change performance much. The dominant resource for MapReduce is memory, so the DRF scheduler continues to balance MapReduce jobs in a manner similar to the default resource calculator. In the case of a single resource, the DRF reduces to max-min fairness for that resource.

Mixed workloads

An example of a mixed workload is a cluster that runs both MapReduce and Storm on YARN. MapReduce is not CPU-constrained, but Storm on YARN is; its containers require more CPU than memory. As you add Storm jobs along with MapReduce jobs, the DRF scheduler tries to balance memory and CPU resources, but you might see some performance degradation in as a result. As you add more CPU-intensive Storm jobs, individual jobs start to take longer to run as the cluster CPU resources are consumed.

To solve this problem, you can use cgroups along with CPU scheduling. Using cgroups provides isolation for CPU-intensive processes such as Storm on YARN, thereby enabling you to predictably plan and constrain the CPU-intensive Storm containers.

You can also use partitions in conjunction with CPU scheduling and cgroups to restrict Storm on YARN jobs to a subset of cluster nodes.

Configure CPU scheduling and isolation

You can configure CPU scheduling on your cluster to allocate the best possible nodes having the required CPU resources for application containers.

Procedure

1. In Cloudera Manager, select the YARN service.
2. Click the Configuration tab.
3. Search for Resource Calculator Class.
4. Select the org.apache.hadoop.yarn.util.resource.DominantResourceCalculator option.
5. Search for yarn.nodemanager.resource.cpu-vcores and set the number of vcores to match the number of physical CPU cores on the NodeManager host by providing the number of physical cores.

Related Information

Limit CPU usage with Cgroups

Using GPU on YARN
Enable Cgroups

Use CPU scheduling with distributed shell

You can run the distributed shell by specifying resources other than memory and vCores. The following is an example for distributed shell but you can use CPU scheduling with other frameworks as well.

Procedure

- Use the following command to allocate two cores for two containers:

```
yarn jar </opt/cloudera/parcels/<CDH-version>/lib/hadoop-yarn/hadoop-yarn-
applications-distributedshell.jar> \
                   -jar </opt/cloudera/parcels/<CDH-version>/lib/hadoop-ya
rn/hadoop-yarn-applications-distributedshell.jar> \
                   -shell_command "sleep 120" \
                   -container_resources memory-mb=3072,vcores=2 \
                   -num_containers 2
```

Use GPU scheduling

On your cluster, you can configure GPU scheduling and isolation. Currently only Nvidia GPUs are supported in YARN. You can useCloudera Manager to configure GPU scheduling on your cluster.

Configure GPU scheduling and isolation

You can configure GPU scheduling and isolation on your cluster. Currently only Nvidia GPUs are supported in YARN.

Before you begin

- YARN NodeManager must be installed with the Nvidia drivers.

Procedure

1. In Cloudera Manager, navigate to Hosts Hosts Configuration .
2. Search for cgroup.
3. Select the Enable Cgroup-based Resource Management checkbox.
4. Click Save Changes.
5. Navigate to YARN Configuration
6. Search for cgroup.
7. Find the Use CGroups for Resource Management property and enable it for the applicable clusters.
8. Find the Always use Linux Container Executor property and enable it for the applicable clusters.
9. Search for gpu.
10. Find the Enable GPU Usage property and select the NodeManager Default Group checkbox.

    Note:

    GPU usage is not supported when using cgroup v2.

11. Find the NodeManager GPU Devices Allowed property and define the GPU devices that are managed by YARN using one of the following ways.
    - Use the default value, auto, for auto detection of all GPU devices. In this case all GPU devices are managed by YARN.
    - Manually define the GPU devices that are managed by YARN.

12. Find the NodeManager GPU Detection Executable property and define the location of nvidia-smi. By default, this property has no value and it means that YARN checks the following paths to find nvidia-smi:
    - /usr/bin
    - /bin
    - /usr/local/nvidia/bin
13. Click Save Changes.
14. Click the Stale Configuration: Restart needed button on the top of the page.
15. Click Restart Stale Services.

    Note that this step restarts all services with stale configurations.
16. Select Re-deploy client configuration and click Restart Now.

If the NodeManager fails to start, the following error is displayed:

```
INFO gpu.GpuDiscoverer (GpuDiscoverer.java:initialize(240)) - Trying to disc
over GPU information ... WARN gpu.GpuDiscoverer (GpuDiscoverer.java:initiali
ze(247)) - Failed to discover GPU information from system, exception message
:ExitCodeException exitCode=12: continue...
```

Fix the error by exporting the LD_LIBRARY_PATH in the yarn -env.sh using the following command:
`export LD_LIBRARY_PATH=/ usr/local/nvidia/lib:/usr/local/nvidia/lib64: $LD_LIBRARY_PATH`

Use GPU scheduling with distributed shell

You can run the distributed shell by specifying resources other than memory and vCores. The following is an example for distributed shell but you can use GPU scheduling with other frameworks as well.

Procedure

- Use the following command to run the distributed shell and GPU without a Docker container:

```
yarn jar </opt/cloudera/parcels/<CDH-version>/lib/hadoop-yarn/hadoop-yarn-
applications-distributedshell.jar> \
                   -jar </opt/cloudera/parcels/<CDH-version>/lib/hadoop-ya
rn/hadoop-yarn-applications-distributedshell.jar> \
                   -shell_command /usr/local/nvidia/bin/nvidia-smi \
                   -container_resources memory-mb=3072,vcores=1,yarn.io/g
pu=2 \
                   -num_containers 2
```

You receive output similar to the following:

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 375.66                 Driver Version: 375.66                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla P100-PCIE...  Off  | 0000:04:00.0     Off |                    0 |
| N/A   30C    P0    24W / 250W |      0MiB / 12193MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla P100-PCIE...  Off  | 0000:82:00.0     Off |                    0 |
| N/A   34C    P0    25W / 250W |      0MiB / 12193MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Use FPGA scheduling

You can use FPGA as a resource type.

A Field Programmable Gate Array (FPGA) is a card that contains a matrix of configurable logic blocks, mostly logic gates. FPGA can be reprogrammed, meaning that the interconnections among the gates and integrated devices can be re-wired. That is in contrast to a regular CPU, as the internal wiring and connections inside a CPU cannot be changed: only the software that runs on it can be changed, but the hardware is always the same. FPGA, on the other hand, allows changes on a circuit level.

An FPGA device does not contain only rewirable logic gates (like AND or OR gates), but also memory, DSP, chip, external connectors and more.

Programming FPGAs

FPGAs can be programmed in various ways. The most popular methods are VHDL and Verilog.

Intel FPGA cards allow the users to upload so-called OpenCL kernels to perform certain computations simultaneously and therefore rapidly. OpenCL is a framework and a set of standards that is currently developed and maintained by the Khronos Group. For more information, see Khronos' OpenCL documentation.

Configure FPGA scheduling and isolation

You can enable and configure FPGA as a resource type using Cloudera Manager.

Before you begin

Ensure that the FPGA Cloudera Runtime or SDK is installed in such a way that the aocl command is executable by the NodeManager. For more information about how to install FPGA, see Intel Acceleration Stack Quick Start Guide for Intel Programmable Acceleration Card with Intel Arria 10 GX FPGA.

Procedure

1. In Cloudera Manager, select the YARN service.
2. Click the Configuration tab.
3. Select FPGA Management category under Filters.
4. Find the Enable FPGA Usage property and select the NodeManagers that can provide FPGA devices to YARN applications that request them.
5. Use the Path to FPGA (aocl) tool property to specify the full local path to the Intel aocl tool installed on the applicable nodes.
6. Use the Allowed FPGA devices to specify the FPGA devices which can be managed by YARN NodeManager.

   Valid values are the following:

   - Auto: Default value. All available FPGA devices are allowed.
   - Comma separated list: List the minor number of the allowed FPGAs. For example: 0,1
7. Use the List of available FPGA devices property to manually specify the available FPGA devices.

   Provide the value in the following format: deviceA/major_number:minor_number,deviceB/major_number:minor_number

   For example: acl0/238:0,acl1/238:1

   The acl numbers are displayed using the aocl diagnose command.

   The major and minor device numbers can usually be determined by listing the device files under /dev. Every card has a corresponding device file. The ls -l command shows both numbers.
8. Use the FPGA device major number property to provide the major device number of the FPGA card.

   This property is used in the container-executor.cfg file.

Important: Only one major number can be specified for each node, meaning that a node can have only one kind of FPGA card.

9. Use the FPGA initializer script property to provide the path to the shell script that sets up the environment for the FPGA device.

   The initializer script sets up various environment variables, and sources other scripts allowing the aocl tool to run properly. The contents of this script depends heavily on the installation. For example, in case of Intel Processing Accelerator Card (PAC), it is necessary to source init_env.sh and init_opencl.sh.

   The following environment variables are possibly required to be exported:

   - AOCL_BOARD_PACKAGE_ROOT
   - CL_CONTEXT_COMPILER_MODE_INTELFPGA

   Please consult the vendor's documentation for further details.
10. Click Save Changes.
11. Restart the affected NodeManagers.

Results

FPGA support is enabled and configured.

What to do next

Request FPGA resource by specifying yarn.io/fpga as a resource.

Use FPGA with distributed shell

Once FPGA support is enabled, an FPGA resource can be requested by specifying yarn.io/fpga as a resource.

The following is a distributed shell example:

```
yarn jar /path/to/hadoop-yarn-applications-distributedshell.jar
-jar /path/to/hadoop-yarn-applications-distributedshell.jar
-shell_command "date"
-container_resources memory-mb=2048,vcores=1,yarn.io/fpga=1
-num_containers 1
```

This command runs the date command on a node which has an FPGA card installed.

Limit CPU usage with Cgroups

You can use cgroups to limit CPU usage in a Hadoop Cluster.

You can use cgroups to isolate CPU-heavy processes in a Hadoop cluster. If you are using CPU Scheduling, you should also use cgroups to constrain and manage CPU processes. If you are not using CPU Scheduling, do not enable cgroups.

When you enable CPU Scheduling, queues are still used to allocate cluster resources, but both CPU and memory are taken into consideration using a scheduler that utilizes Dominant Resource Fairness (DRF). In the DRF model, resource allocation takes into account the dominant resource required by a process. CPU-heavy processes (such as Storm-on-YARN) receive more CPU and less memory. Memory-heavy processes (such as MapReduce) receive more memory and less CPU. The DRF scheduler is designed to fairly distribute memory and CPU resources among different types of processes in a mixed- workload cluster.

Cgroups compliments CPU Scheduling by providing CPU resource isolation. It enables you to set limits on the amount of CPU resources granted to individual YARN containers, and also lets you set a limit on the total amount of CPU resources used by YARN processes.

Cgroups represents one aspect of YARN resource management capabilities that includes CPU Scheduling, partitions, archival storage, and memory as storage. If CPU Scheduling is used, cgroups should be used along with it to constrain and manage CPU processes.

Related Information

Configure CPU scheduling and isolation

Use Cgroups

You can use strict cgroups CPU limits to constrain CPU processes in mixed workload clusters.

One example of a mixed workload is a cluster that runs both MapReduce and Storm-on-YARN. MapReduce is not CPU-constrained (MapReduce containers do not ask for much CPU). Storm-on-YARN is CPU-constrained: its containers ask for more CPU than memory. As you start adding Storm jobs along with MapReduce jobs, the DRF scheduler attempts to balance memory and CPU resources, but as more CPU-intensive Storm jobs are added, they may begin to take up the majority of the cluster CPU resources.

You can use cgroups along with CPU scheduling to help manage mixed workloads. cgroups provide isolation for CPU-heavy processes such as Storm-on-YARN, thereby enabling you to predictably plan and constrain the CPU-intensive Storm containers.

When you enable strict cgroup CPU limits, each resource gets only what it asks for, even if there is extra CPU available. This is useful for scenarios involving charge-backs or strict SLA enforcement, where you always need to know exactly what percentage or CPU is being used.

Also, enabling strict CPU limits would make job performance predictable, whereas without setting strict limits a CPU-intensive job would run faster when the cluster was not under heavy use, but slower when more jobs were running in the cluster. Strict CPU limits would therefore also be useful for benchmarking.

You can also use partitions in conjunction with cgroups and CPU scheduling to restrict Storm-on-YARN jobs to a subset of cluster nodes.

If you are using cgroups and want more information on CPU performance, you can review the statistics available in the /cgroup/cpu/yarn/cpu.stat file.

Enable Cgroups

You can enable CPU Scheduling to enable cgroups. You must configure certain properties in yarn-site.xml on the ResourceManager and NodeManager hosts to enable cgroups.

About this task

cgroups is a Linux kernel feature. cgroups is supported on the following Linux operating systems:

- CentOS
- RHEL
- SUSE
- Ubuntu



Important:

The yarn.nodemanager.linux-container-executor.cgroups.mount property must be set to false. Setting this value to true is not currently supported.

Procedure

Enable cgroups

The following commands must be run on every reboot of the NodeManager hosts to set up the cgroup hierarchy. Note that operating systems use different mount points for the cgroup interface. Replace /sys/fs/cgroup with your operating system equivalent.

```
mkdir -p /sys/fs/cgroup/cpu/yarn
```

```
chown -R yarn /sys/fs/cgroup/cpu/yarn
mkdir -p /sys/fs/cgroup/memory/yarn
chown -R yarn /sys/fs/cgroup/memory/yarn
mkdir -p /sys/fs/cgroup/blkio/yarn
chown -R yarn /sys/fs/cgroup/blkio/yarn
mkdir -p /sys/fs/cgroup/net_cls/yarn
chown -R yarn /sys/fs/cgroup/net_cls/yarn
mkdir -p /sys/fs/cgroup/devices/yarn
chown -R yarn /sys/fs/cgroup/devices/yarn
```

1. In Cloudera Manager, select the YARN service.
2. Click the Configuration tab.
3. Search for Always Use Linux Container Executor and select YARN-1 (Service-Wide) option.
4. Search for NodeManager Advanced Configuration and set the below property in the NodeManager Advanced Configuration Snippet (Safety Valve) for yarn-site.xml field.

   ```
   Name:  yarn.nodemanager.linux-container-executor.group
   Value: hadoop
   yarn.nodemanager.linux-container-executor.cgroups.v2.enabled=true
   yarn.nodemanager.linux-container-executor.cgroups.mount-path=/sys/fs/cgro
   up
   ```

5. Search for CGroups and select YARN-1 (Service-Wide) option in the Use CGroups for Resource Management field.
6. Search for CGroups Hierarchy and set the NodeManager Default Group value to /hadoop-yarn.
7. Search for NodeManager Advanced Configuration and set the below property in the NodeManager Advanced Configuration Snippet (Safety Valve) for yarn-site.xml field.

   ```
   Name:  yarn.nodemanager.linux-container-executor.cgroups.mount
   Value: false
   ```

   ```
   Name:  yarn.nodemanager.linux-container-executor.cgroups.mount-path
   Value: /sys/fs/cgroup
   ```

8. (Optional) Set the percentage of CPU to be used by YARN. Search for Containers CPU Limit and set the value in the Containers CPU Limit Percentage field.

   Set the percentage of CPU that can be allocated for YARN containers. In most cases, the default value of 100% should be used. If you have another process that needs to run on a node that also requires CPU resources, you can lower the percentage of CPU allocated to YARN to free up resources for the other process.
9. (Optional) Set flexible or strict CPU limits. Search for Strict CGroup Resource Usage and select the NodeManager Default Group field.

   CPU jobs are constrained with CPU scheduling and cgroups enabled, but by default these are flexible limits. If spare CPU cycles are available, containers are allowed to exceed the CPU limits set for them. With flexible limits, the amount of CPU resources available for containers to use can vary based on cluster usage -- the amount of CPU available in the cluster at any given time.

   You can use cgroups to set strict limits on CPU usage. When strict limits are enabled, each process receives only the amount of CPU resources it requests. With strict limits, a CPU process will receive the same amount of cluster resources every time it runs.

   Strict limits are not enabled (set to false) by default.

   Note: Irrespective of whether this property is true or false, at no point will total container CPU usage exceed the limit set in yarn.nodemanager.resource.percentage-physical-cpu-limit.

Note: CPU resource isolation leverages advanced features in the Linux kernel. At this time, setting yarn.nodemanager.linux-container-executor.cgroups.strict-resource-usage to true is not recommended due to known kernel panics. In addition, with some kernels, setting yarn.nodemanager.resource.percentage-physical-cpu-limit to a value less than 100 can result in kernel panics. If you require either of these features, you must perform scale testing to determine if the in-use kernel and workloads are stable. As a starting point, Linux kernel version 4.8.1 works with these features. However, testing the features with the desired workloads is very important.

Related Information

Configure CPU scheduling and isolation

Cloudera Support Matrix

Managing YARN Queue Manager

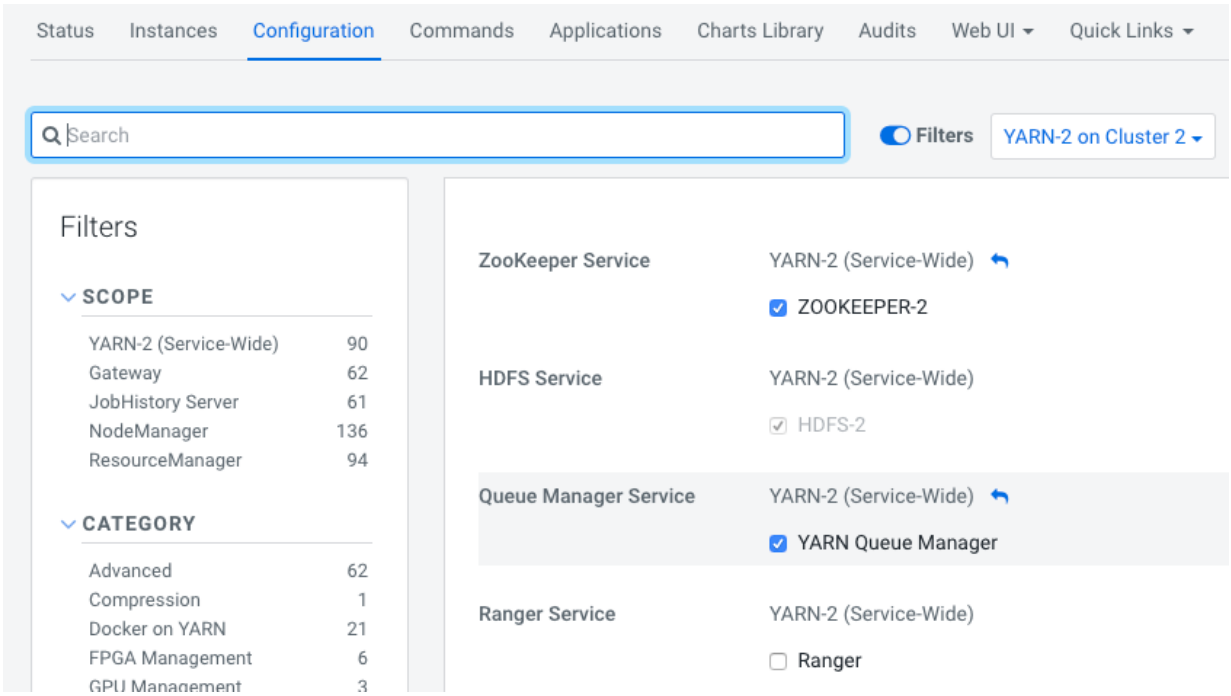
YARN Queue Manager manages Capacity Scheduler configuration It also provides a user interface (YARN Queue Manager UI) that makes it easier for Hadoop operators to create, configure and manage YARN queues on a global or queue level.

As Capacity Scheduler is the supported scheduler in Cloudera, it is recommended to use YARN Queue Manager to manage the scheduler configuration.

Configuring YARN Queue Manager dependency

If you add the YARN Queue Manager service to the cluster after installing the cluster, you must configure the YARN Queue Manager dependency.

1. In Cloudera Manager, select the YARN service.
2. Click Configuration.
3. Search for Queue Manager service.
4. Select the YARN Queue Manager checkbox.



5. Click Save Changes.

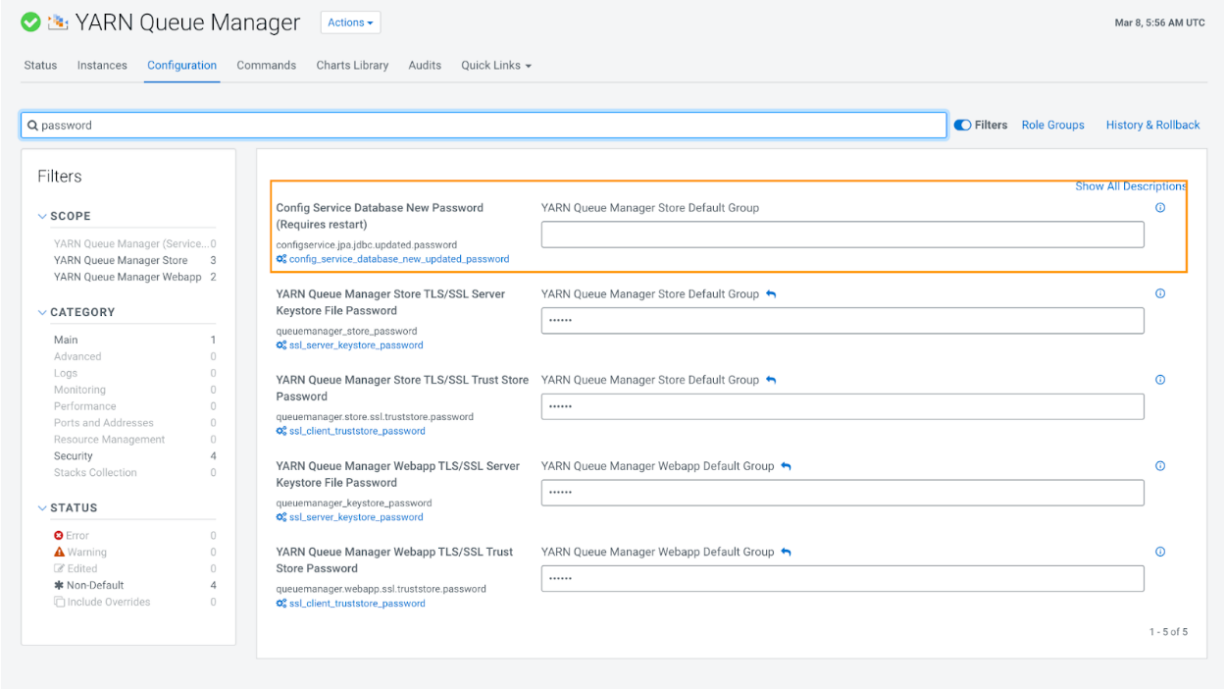
6. Restart the YARN and the YARN Queue Manager services.

Updating YARN Queue Manager Database Password

Once you have configured Queue Manager, you can update the YARN Queue Manager database password manually.

Procedure

1. In Cloudera Manager navigate to Clusters YARN Queue Manager UI
2. Click Configuration.
3. Search for password.

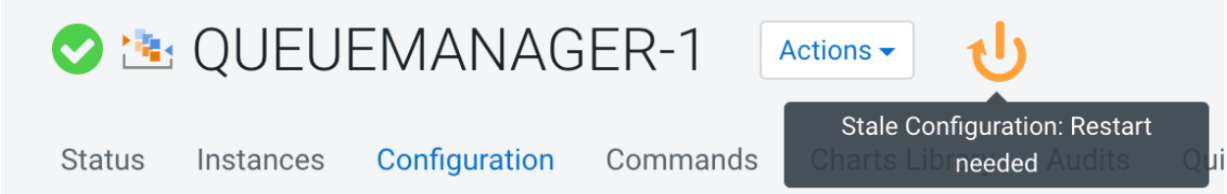


The Config Service Database New Password (Requires restart) field is displayed.

4. Enter the new password into the input field.

Note: Do not use the # symbol in your password.

5. Click Save Changes.
6. Restart the YARN Queue Manager service.
   On restarting, the password of the database is updated to the new password.
7. On restarting YARN Queue Manager, a stale configuration icon is displayed.



8. Restart the YARN Queue Manager service again to address this.

Accessing the YARN Queue Manager UI

Once you have enabled the Queue Manager UI, you can use it to manage your scheduler configuration.

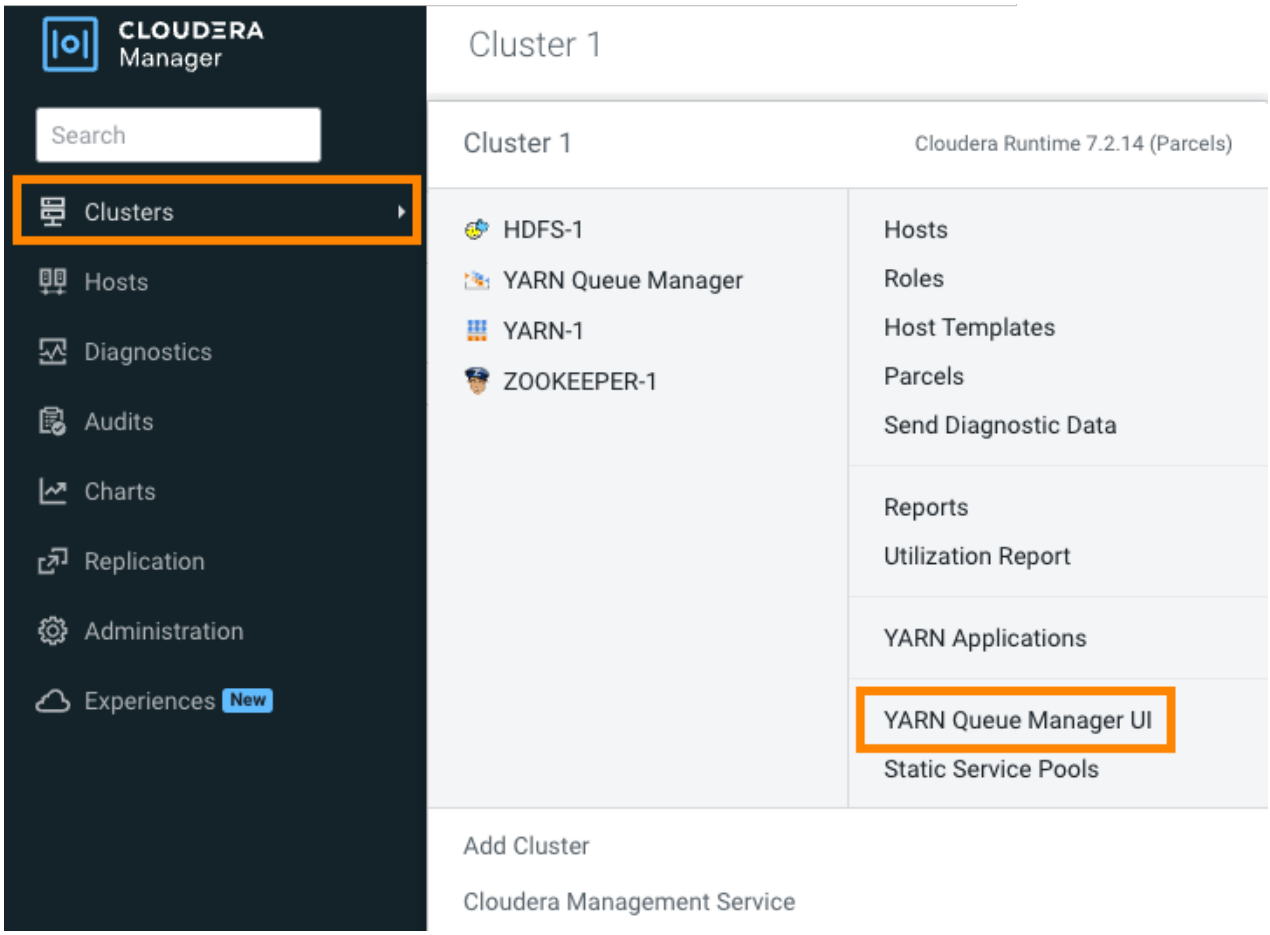
About this task

Before you begin

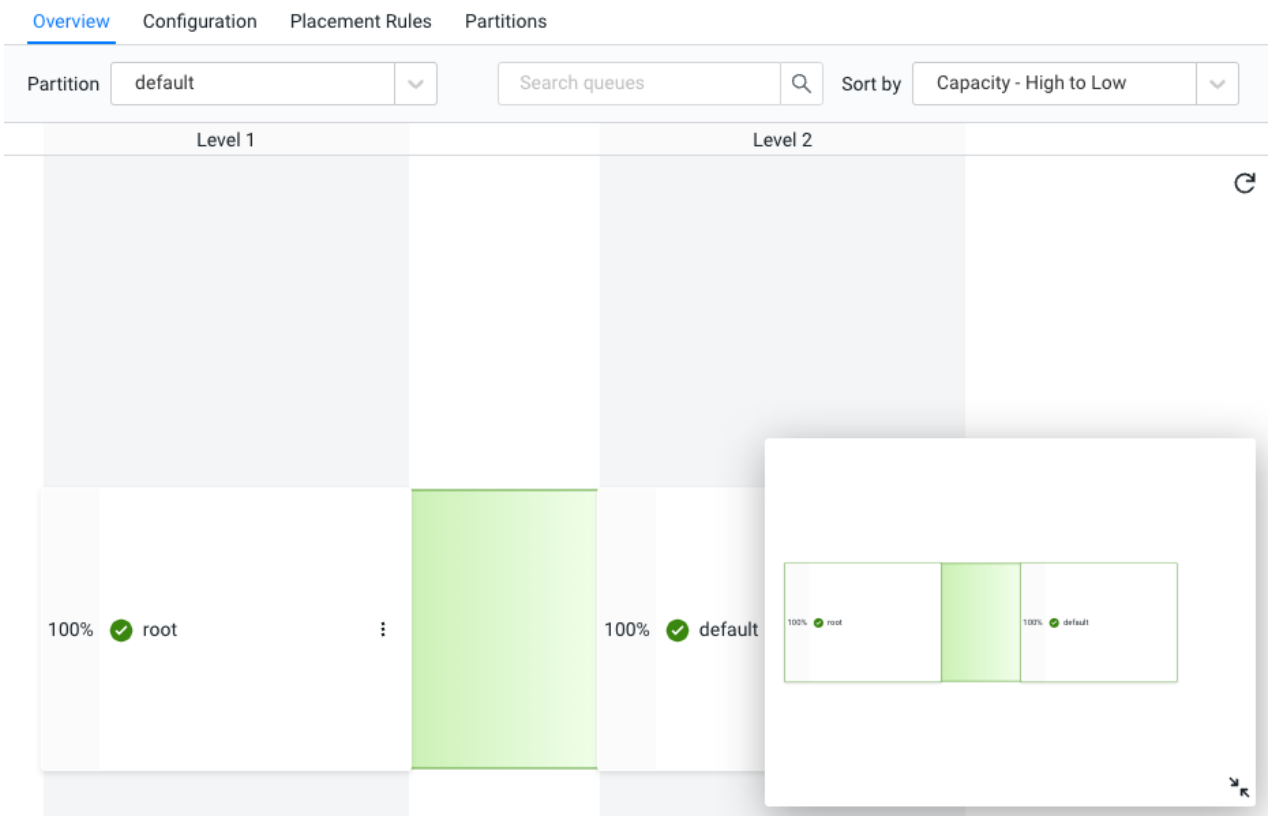
- Add the YARN Queue Manager service to your cluster
- Configure the YARN Queue Manager dependency

Procedure

In Cloudera Manager navigate to Clusters YARN Queue Manager UI



The Overview tab is opened, displaying a hierarchical view of the queues:



What to do next

Navigate between the difference tabs to manage queues, configure queue and global level properties, and to manage YARN features.

Related Information

Managing queues

Configuring scheduler properties at the global level

Configuring per queue properties

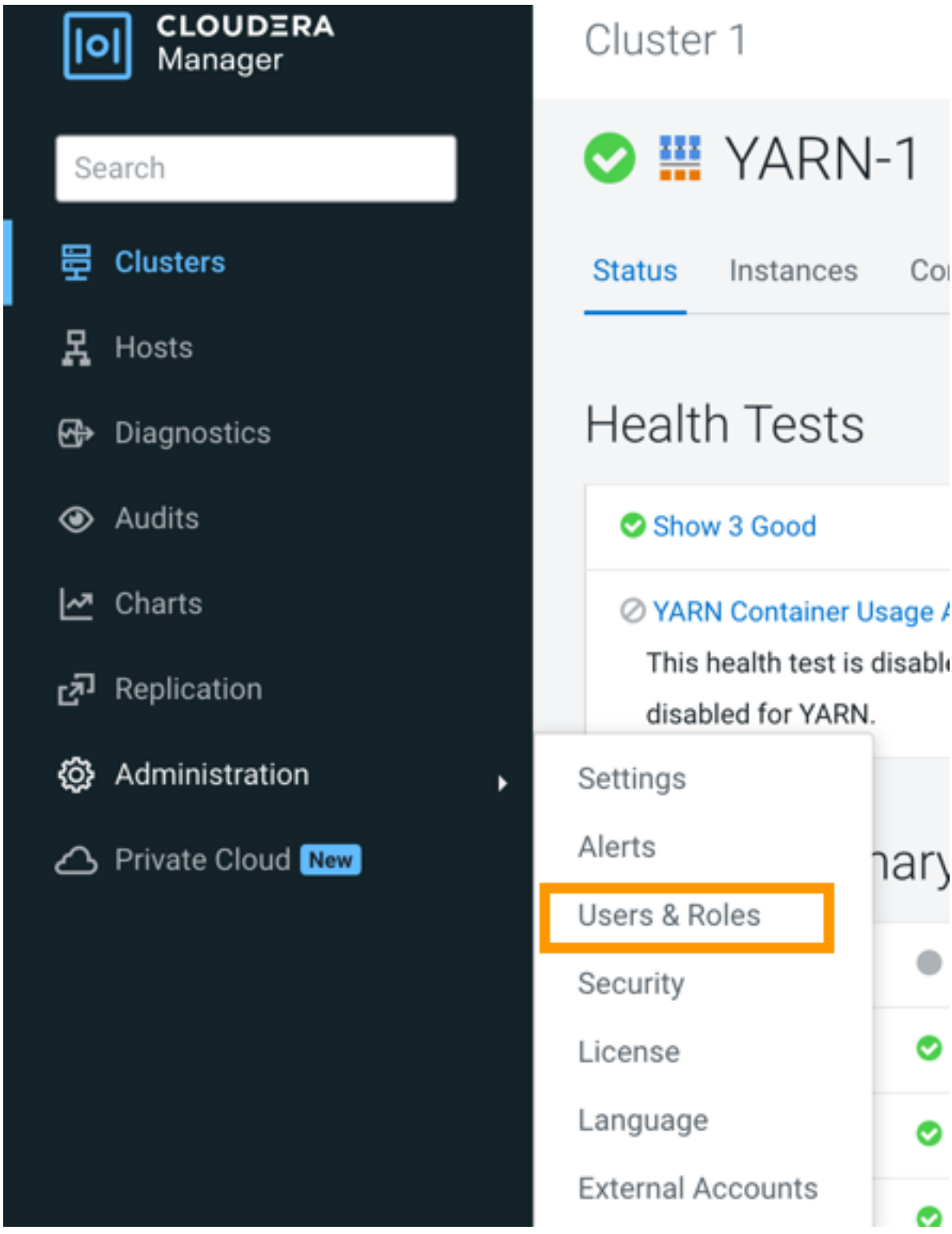
Managing placement rules

Partition configuration

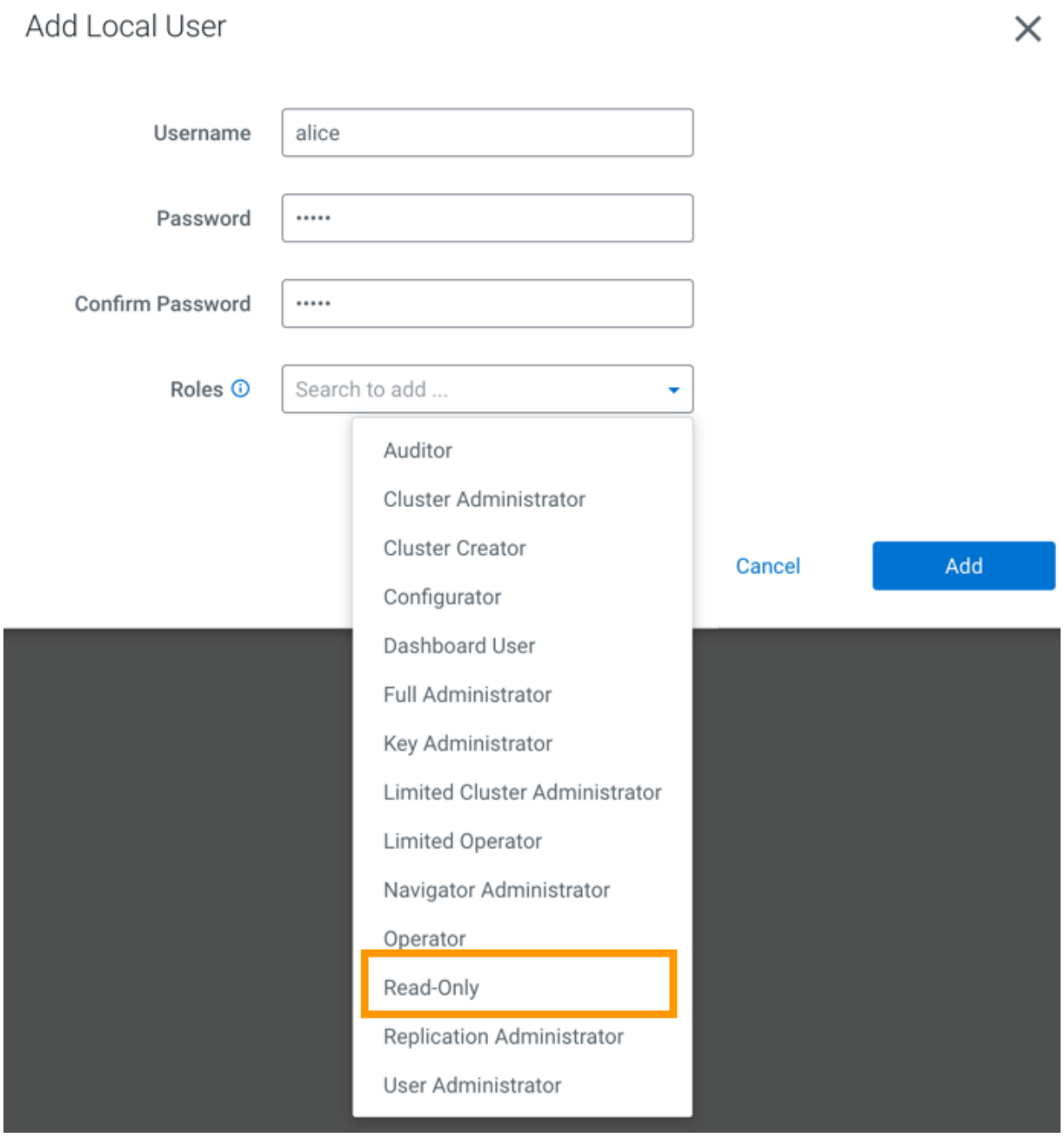
Providing read-only access to Queue Manager UI

You can now allow non-admin users to access YARN Queue Manager in a read-only mode. You can either create a new user account with read-only role or use any existing user account with read-only role in Cloudera Manager to access YARN Queue Manager UI. In the read-only access mode, the user can view all the configurations but cannot make any changes to the configurations.

1. In Cloudera Manager, click the Administration > Users & Roles.

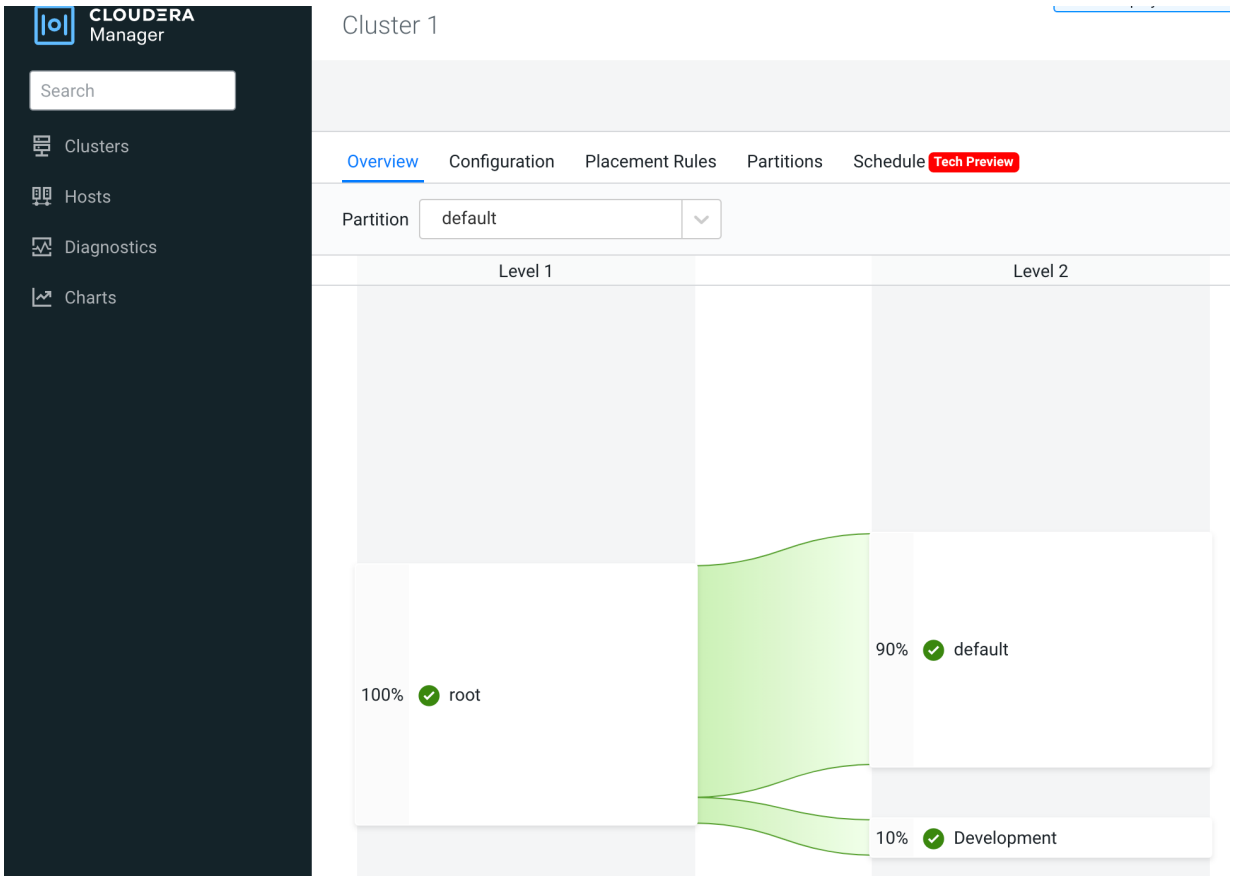


2. Provide username, password, and select Read-Only from the Roles dropdown list.



3. Click Add.

   For more information about assigning user roles, see Cloudera Manager User Roles.
4. Click on your username on the left navigation pane and select Sign Out.
5. Log in to Cloudera Manager as the newly created Read Only user.
6. Click Clusters > YARN Queue Manager UI service. The YARN Queue Manager UI is displayed without edit options.



Configuring the embedded Jetty Server in Queue Manager

The configuration properties of the Jetty Server embedded in the Queue Manager can be configured using Cloudera Manager. The configuration of these properties can facilitate debugging.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager service.
2. Click Configuration.
3. Search for Jetty property to list all the available Jetty Server configuration properties.

   For a comprehensive list of Jetty Server configuration properties, see *YARN Queue Manager properties reference* in the Cloudera Manager documentation.
4. Configure the Jetty Server configuration properties according to your use case.
5. Click Save Changes.
6. Restart the YARN Queue Manager service.

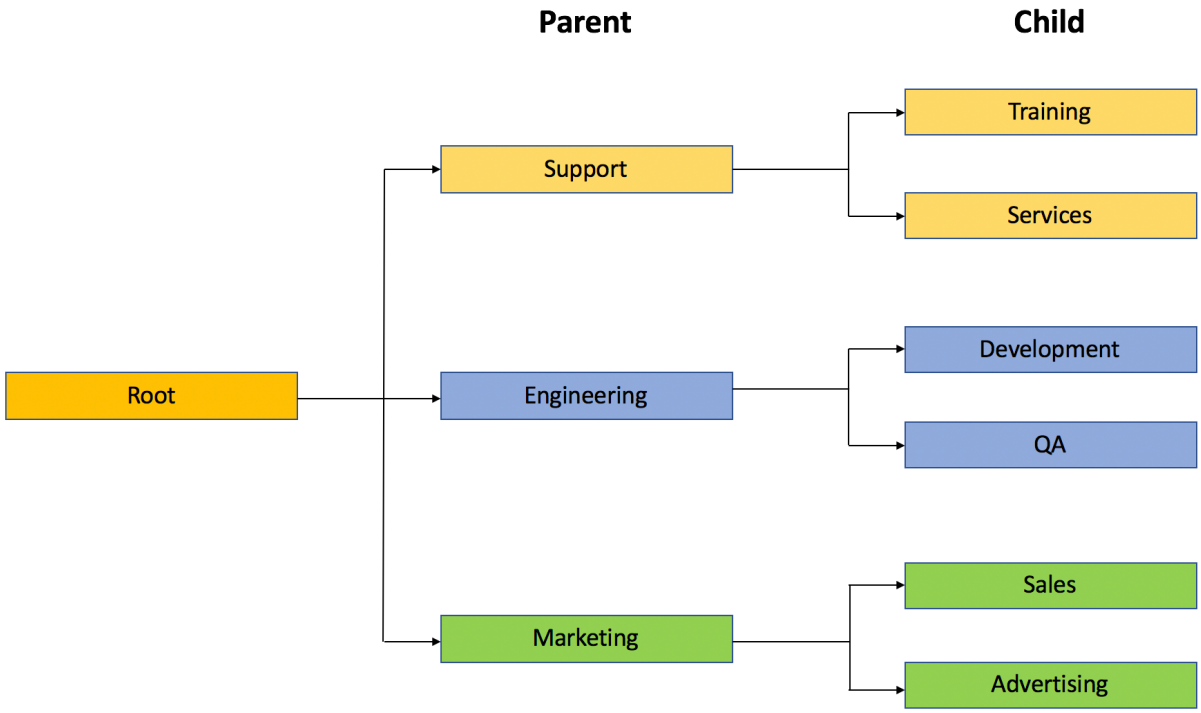
Managing queues

YARN Queue Manager is the queue management graphical user interface for Apache Hadoop YARN Capacity Scheduler. You can use YARN Queue Manager UI to manage your cluster capacity using queues to balance resource requirements of multiple applications from various users. Using YARN Queue Manager UI, you can set scheduler level properties and queue level properties.

You can view, sort, search, and filter queues using YARN Queue Manager UI. Queue Manager stores history of previous changes and provides the ability to view the changes of each version in the Overview and Scheduler Configuration tabs. The previous versions will be in the read-only mode and you must select the latest version to make changes.

The fundamental unit of scheduling in YARN is a queue. The capacity of each queue specifies the percentage of cluster resources that are available for applications submitted to the queue. Capacity Scheduler queues can be set up in a hierarchy that reflects the database structure, resource requirements, and access restrictions required by the various organizations, groups, and users that utilize cluster resources.

For example, suppose that a company has three organizations: Engineering, Support, and Marketing. The Engineering organization has two sub-teams: Development and QA. The Support organization has two sub-teams: Training and Services. And finally, the Marketing organization is divided into Sales and Advertising. The following image shows the queue hierarchy for this example:



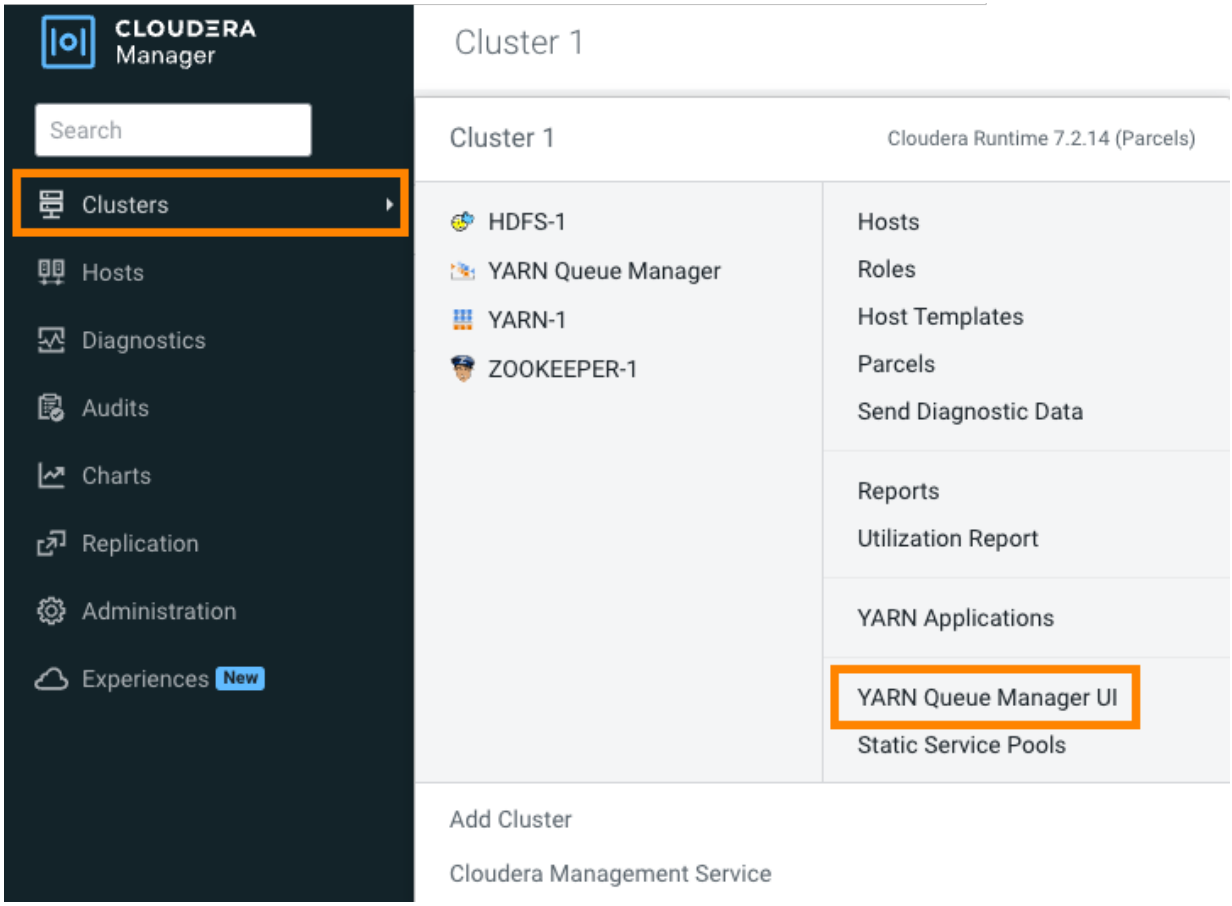
Each child queue is tied to its parent queue and the top-level "support", "engineering", and "marketing" queues would be tied to the "root" queue.

Adding queues using YARN Queue Manager UI

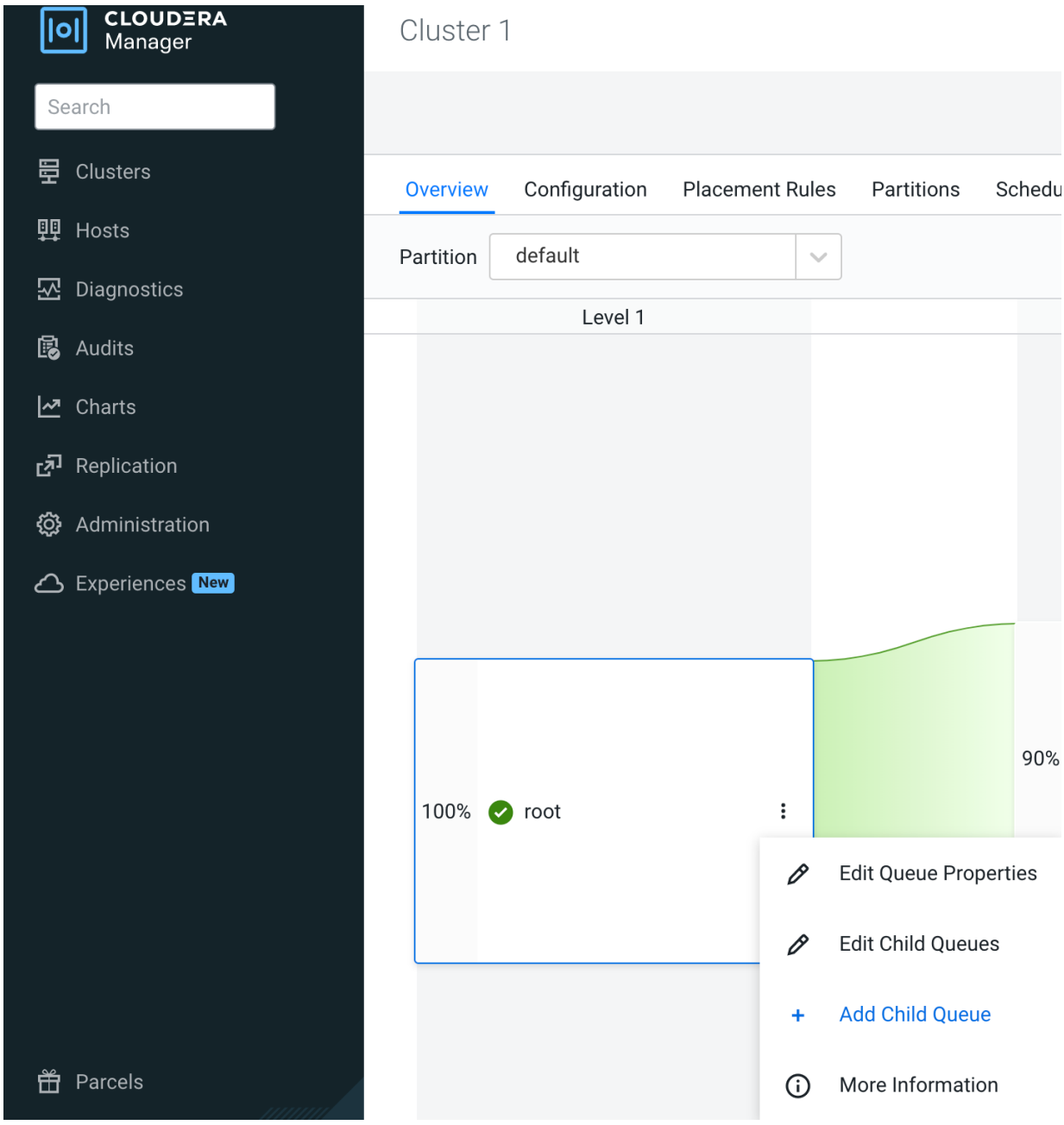
You can add queues to the predefined queue called root from the Yarn Queue Manager UI. The Capacity Scheduler has a predefined queue called root. All queues in the system are children of the root queue. Each child queue is tied to its parent queue, but children queues do not inherit properties directly from the parent queue unless otherwise specified.

Procedure

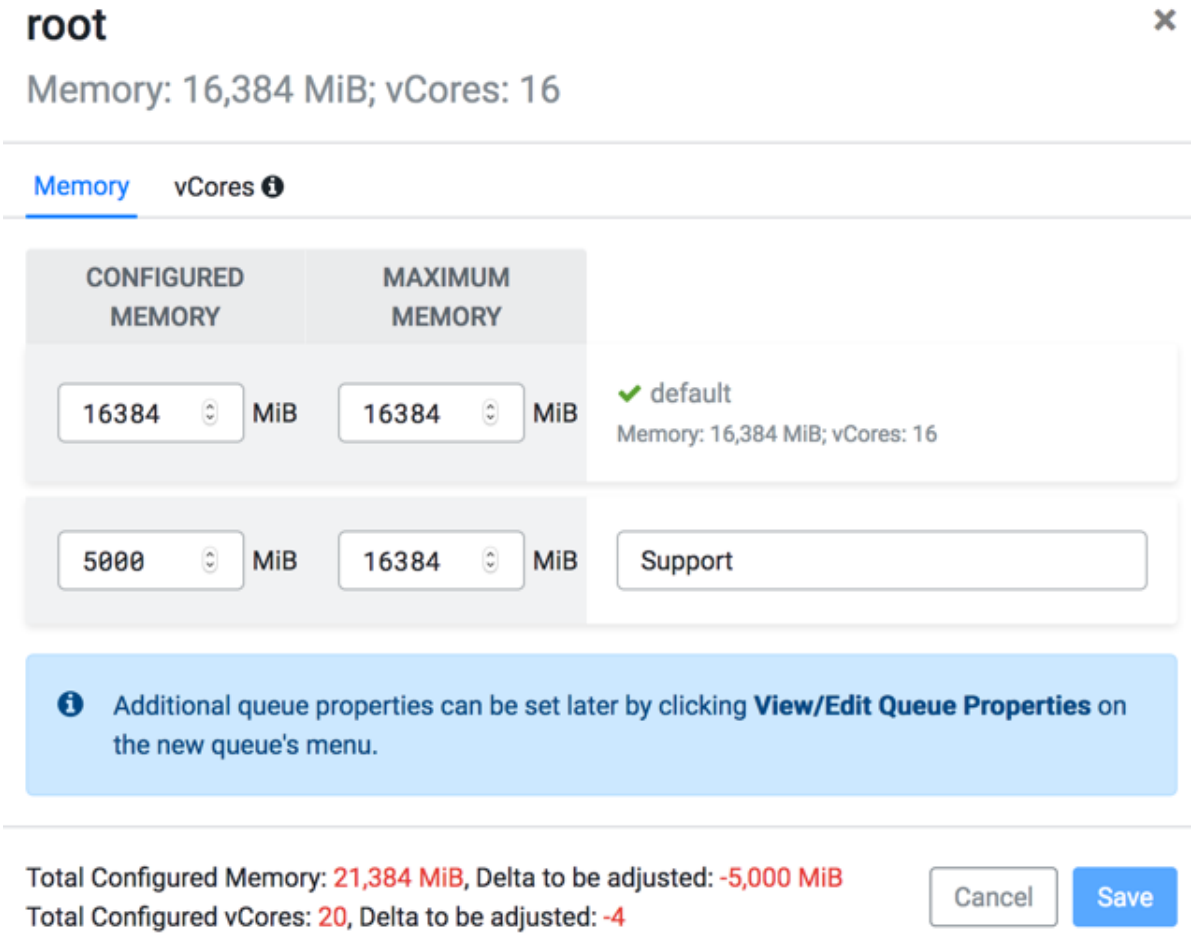
1. In Cloudera Manager, navigate to Clusters YARN Queue Manager UI .



2. Click on the three vertical dots on the root and select the Add Child Queue option.

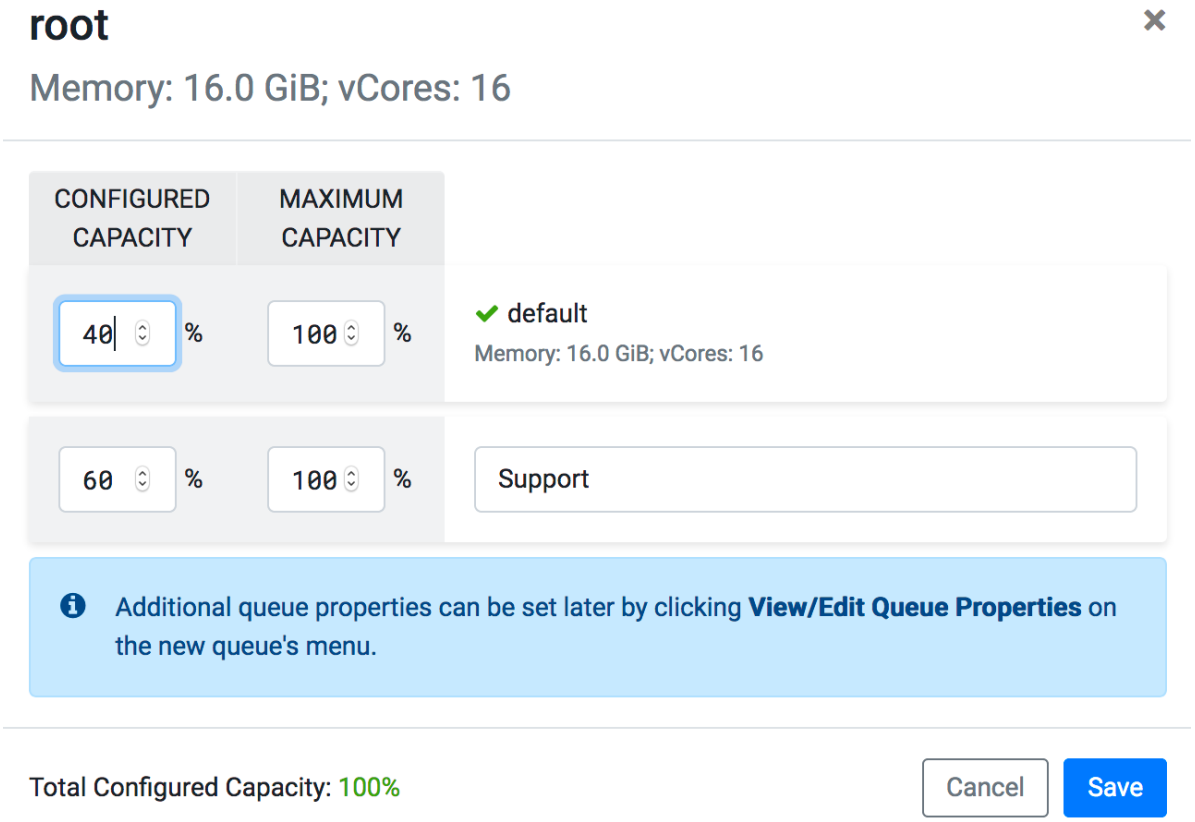


3. Enter the information based on the Relative or Absolute or Weight allocation mode.
   - Absolute allocation mode: Enter the name of the queue and units of memory in MiB in the Memory tab. Enter the number of cores in the vCores tab.

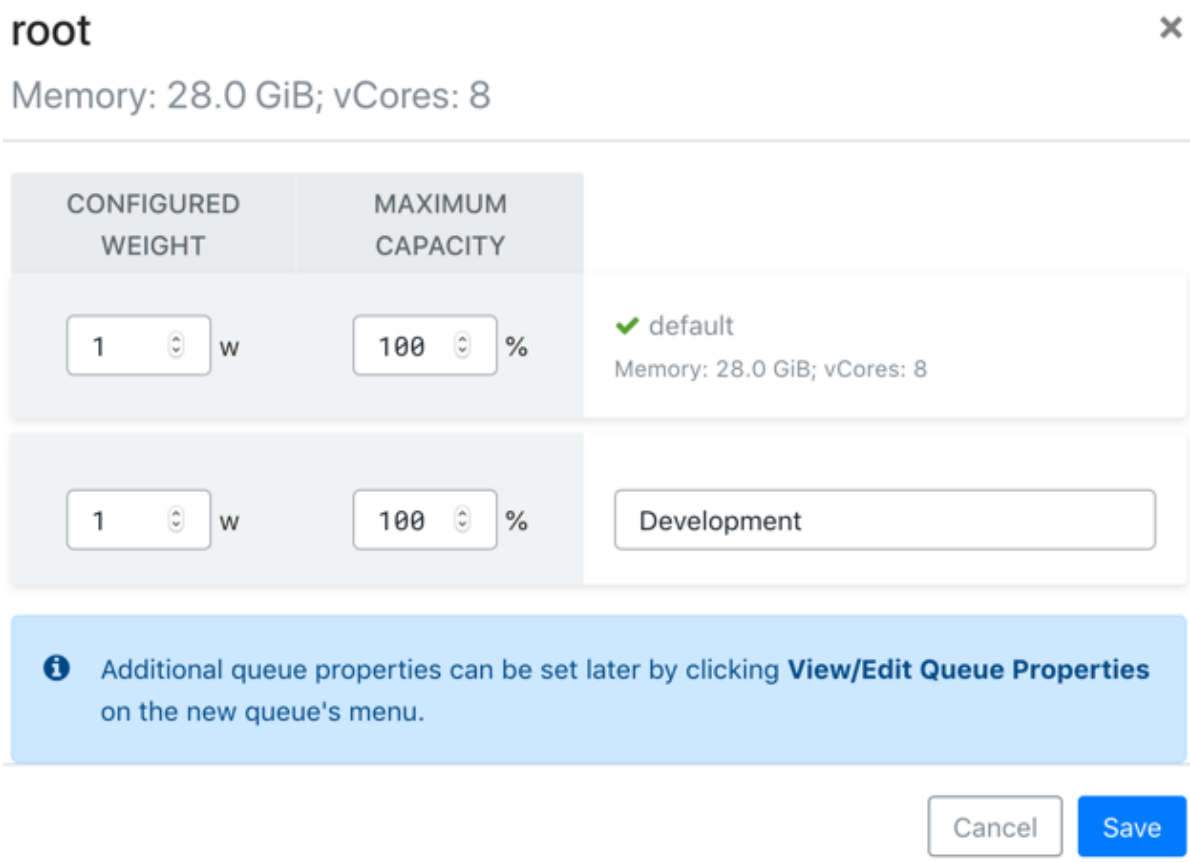


Note: When decreasing a queue's configured resource capacity to allocate resources for the new sibling queue you must update resource capacities starting from the lowest queue level. This is because a parent queue's capacity cannot be decreased until the total resource capacity of all of its child queues is reduced. You might have to change both the minimum and maximum values.

- Relative allocation mode: Enter the name of the queue, Configured Capacity, and Maximum Capacity values for the queue.



- Weight allocation mode: Enter the name of the queue and the fraction of the resource of the resource in the Configured Weight for the queue.



4. Click Save.

   You can continue to add more parent and child queues by following the same steps.

Related Information

Resource allocation overview

Changing resource allocation mode

Configuring cluster capacity with queues

You can manage your cluster capacity using queues to balance resource requirements of multiple applications from various users.

You can use the Capacity Scheduler to share cluster resources using FIFO (first in, first out) queues.

You can configure queues either by specifying the percentage of the capacity using the Relative mode or the actual units of vCores and memory using the Absolute mode or the fraction of the total capacity using the Weight mode. If you are upgrading your cluster, Weight mode is the default mode. If you are installing and configuring the cluster freshly, Relative mode is the default mode. You can change the allocation mode to the Absolute mode.

You can submit applications to different queues at multiple levels in the queue hierarchy if the capacity is available on the nodes in the cluster. Because total cluster capacity can vary, capacity configuration values are expressed using percentages, units, or fractions.

Example – Configuring capacity using the Relative mode

You can specify the capacity property to allocate a floating-point percentage value of cluster capacity to a queue. The following configurations divide the cluster resources between the "engineering", "support", and "marketing" organizations in a 6:1:3 ratio (60%, 10%, and 30%).

To specify the capacity property based on the above example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the root and select Edit Child Queues option.
3. Enter the Configured Capacity of "engineering" to 60, "support" to 10, and "marketing" to 30.
4. Click Save.

Example – Configuring capacity using the Absolute mode

You can specify the capacity in units of memory in MiB and the number of cores to a queue. If the total units of memory is 16384 MiB and 16 cores, and the the cluster resources between the "engineering", "support", and "marketing" organizations are allocated as follows:

Organization	Memory	vCores
engineering	9830	10
support	1638	2
marketing	4916	4

To specify the capacity property based on the above example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the root and select Edit Child Queues option.
3. Enter the Configured Memory of "engineering" to 9830, "support" to 1638, and "marketing" to 4916.
4. Enter the Configured vCores of "engineering" to 10, "support" to 2, and "marketing" to 4.
5. Click Save.

Example – Configuring capacity using the Weight mode

You can specify the capacity in fractions of total resources. The resources are divided based on how the queue's weight relates to the sum of configured weights under the parent. The following configurations divide the cluster resources between the "engineering", "support", and "marketing" organizations in 6:1:3 (6/10, 1/10, and 3/10) fraction of total resources.

To specify the capacity property based on the above example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the root and select Edit Child Queues option.
3. Enter the Configured Weight of "engineering" to 6, "support" to 1, and "marketing" to 3.
4. Click Save.

Maximum Capacity: The Maximum Capacity is to the right field or fields after selecting Edit Child Queues in the Queue Manager UI for any queue. The name of the Maximum Capacity varies depending on the root queue property for Absolute, Relative or Weighted. The purpose of the Maximum Capacity is to allow queues to borrow resources from other queues when they are not in use and when they are suitable for the application and queue. A common use case for this is, a queue used for ad-hoc queries or low priority data sicence experiements which a user would like to borrow resources and are fine with loosing them at any point. Usually, it is a best practice to set the configured capacity equal to maximum capacity when a queue has interqueue preemption disabled otherwise the queue may tie up borrowed resources which are garunteed to another queue's configured capacity.

Related Information

Resource allocation overview

Configuring the resource capacity of root queue in absolute mode

In absolute resource allocation mode when you remove or add new resources to your cluster, you can update the resource capacity for the root queue accordingly. You can configure the capacity of the root queue using the YARN Queue Manager UI.

About this task

In absolute resource allocation mode when the available resource capacity changes permanently in the cluster or in a partition, reconfiguring the resource capacity of the root queue might be needed.

In case of a resource capacity increase, you can update the root queue directly.

However, if there was a decrease in the available resource capacity, you cannot update the root queue until child queues have been updated. In a case like this, you must update resource capacities starting from the lowest queue level. This is because a parent queue's capacity cannot be decreased until the total resource capacity of all of its child queue is reduced.

Procedure

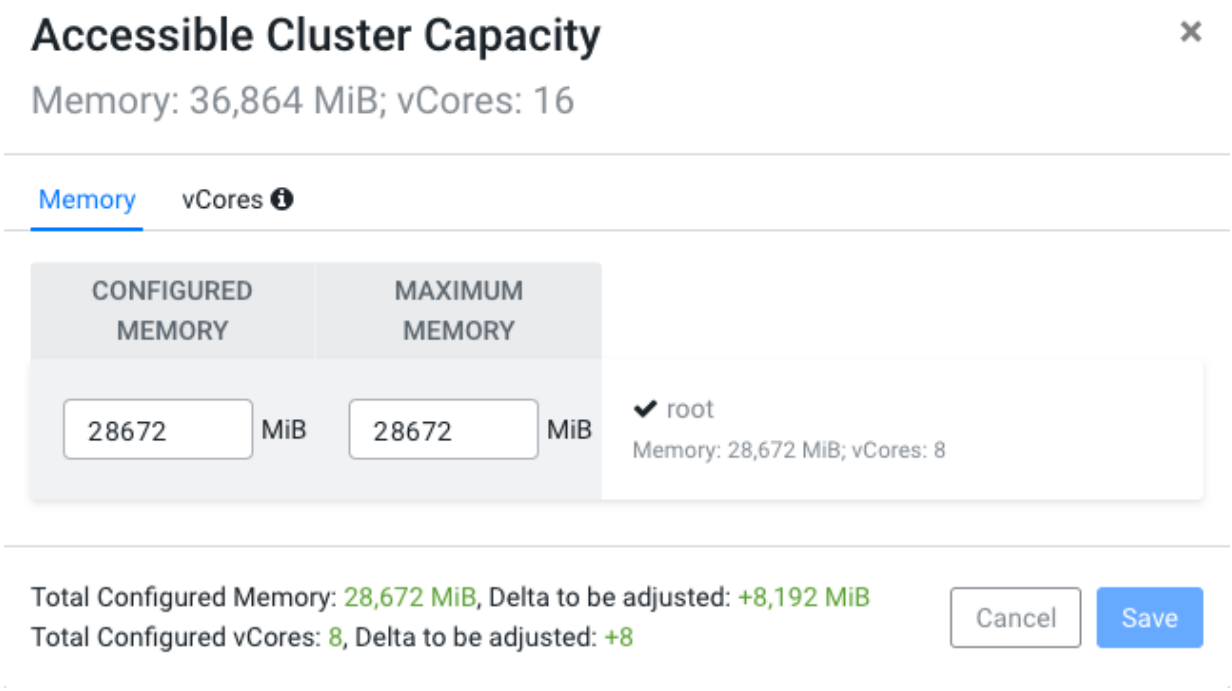
1. In Cloudera Manager navigate to Clusters YARN Queue Manager UI .
   The **Overview** tab is opened, displaying a hierarchical view of the queues.
2. If there was a decrease in the available resources, configure the resource capacity of the child queues.

   You must update resource capacities starting from the lowest queue level. This is because a parent queue's capacity cannot be decreased until the total resource capacity of all of its child queue is reduced. You might have to change both the minimum and maximum values.
3. Click the three vertical dots next to the root queue.
4. Select Edit Root Queue.
   The **Accessible Cluster Capacity** dialog box is displayed.
5. Configure resource capacity for the root queue.

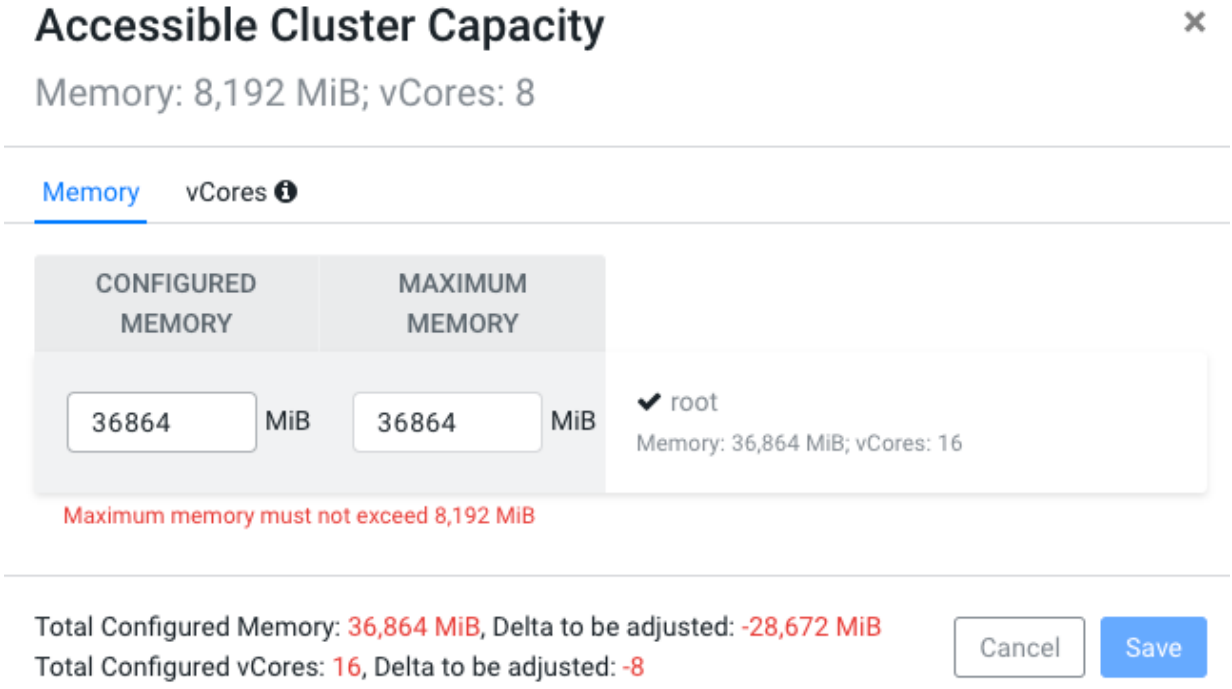
   The **Accessible Cluster Capacity** dialog box provides information about the available resources:

   - Total Configured Memory and the delta to be adjusted
   - Total Configured vCores and the delta to be adjusted

   If there was an increase in the resource capacity, the **Accessible Cluster Capacity** dialog box looks similar to the following:



If there was a decrease in the resource capacity, the **Accessible Cluster Capacity** dialog box looks similar to the following:



6. Click Save.

   If the adjustable delta is less than zero, you cannot save the updated configuration.

After saving the resource capacity of the root queue, the **Failed to update queue configuration** error message is displayed with the following details: Validation failed for modify queue operation. Error message: CapacityScheduler configuration validation failed:java.io.IOException: Failed to re-init queues : Parent Queues capacity: <memory:*[***CONFIGURED MEMORY OF THE ROOT QUEUE***]*, vCores:*[***CONFIGURED VCORES OF*

*THE ROOT QUEUE****]*> is less than to its children:<memory:*[****CONFIGURED MEMORY OF THE CHILD QUEUES****]*, vCores:*[****CONFIGURED VCORES OF THE CHILD QUEUES****]*> for queue:root

This error message appears when there was a decrease in the available resources and you updated the resource capacity of the root queue before updating the child queues.

To resolve this issue, update the resource capacity of the child queues first, and then update the root queue.

Mixed resource allocation mode (Technical Preview)

In this mode, you can specify the resources in mixed types. You can specify the actual units of vcores and memory resources for each queue or specify the percentage of the total resources used by each queue or specify a weight for each queue. You can use a combination of these allocation modes. The queues under one parent can also mix their modes.

Technical Preview: This is a technical preview feature and is considered under development. Do not use this in your production systems. To share your feedback, contact Support by logging a case on our Cloudera Support Portal. Technical preview features are not guaranteed troubleshooting guidance and fixes.

Note: YARN Queue Manager does not support mixed resource allocation mode. This feature is only available through the safety valve.

Example of mixed mode resource allocation calculation

Capacity types are evaluated in the following order (precedence):

1. Absolute value
2. Percentage value
3. Weight value

An example of a calculation of queues *A*, *B*, and *C* with the following configured capacity vectors all resources: [memory=16284, vcore=16, custom=100]

- a: [memory=4096, vcore=50%, custom=3w]
- b: [memory=30%, vcore=10, custom=5w]
- c: [memory=70%, vcore=1w, custom=50]

The calculation of the effective resource is evaluated as the following:

1. Memory: *A* has an absolute resource, and *B* and *C* have a percentage, therefore the percentage is calculated from the remaining resource after allocating queue *A*. In this example, the memory of the a queue is 4096. In the remaining resources (16284 - 4096 = 12188), *B* queue gets 8532 (70% of 12188) and *C* queue gets 3656 (30% of 12188).
2. Vcore: *B* has an absolute resource, *A* has a percentage, then *C* has weight, which gets the remaining vcore. In this example, the vcore of the *B* queue is 10. In the remaining vcores (16 - 10 = 6), *C* queue gets 3 (1W of 6) and *A* queue gets 3 (50% of 6).
3. Custom: *C* has an absolute resource, and *B* and *C* have weight. In this example, the custom of the *C* queue is 50. In the remaining vcores (100 - 50 = 50), *B* queue gets 15 (5w of 50) and *A* queue gets 35 (3w of 50).

Related Information

Cloudera Support Portal

Setting capacity using mixed resource allocation mode (Technical Preview)

Set the resource allocation in the mixed allocation mode using Cloudera Manager.

Procedure

1. In Cloudera Manager, select the YARN service.
2. Go to the Configuration tab.

3. Set the following property in the Capacity Scheduler Configuration Advanced Configuration Snippet (Safety Valve) field to use the mixed allocation mode:

```
<property>
<name>yarn.scheduler.capacity.legacy-queue-mode.enabled</name>
<value>false</value>
</property>
```

4. Specify the capacity for each queue in the preferred allocation mode. Example:

```
<property>
<name>yarn.scheduler.capacity.root.queues</name>
<value>default,test_1,test_2</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.capacity</name>
<value>100</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.default.capacity</name>
<value>[memory=25%, vcores=1w]</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.test_1.capacity</name>
<value>[memory=1w, vcores=10%]</value>
</property>
<property>
<name>yarn.scheduler.capacity.root.test_2.capacity</name>
<value>[memory=2048, vcores=2w]</value>
</property>
```

5. Click Save Changes.
6. Click Actions > Restart to restart the service.

Related Information

Resource allocation overview

Changing resource allocation mode

There are three supported resource allocation modes: absolute, relative, and weight resource allocation mode. When creating a new cluster, the default allocation mode is the relative resource allocation mode. You can change the resource allocation mode from the root queue by editing the queue properties using the Yarn Queue Manager UI.

About this task

The way you specify how resources are allocated is done differently in each of the resource allocation modes:

- Absolute resource allocation mode: In this mode, you specify the actual units of vcores and memory resources. It allows you to configure minimum resources independently for each queue.
- Relative resource allocation mode: In this mode, you specify the percentage of the total resources used by each queue.
- Weight resource allocation mode: In this mode, you specify a weight to each queue. For example, a queue with weight 2 should receive approximately twice as many resources as a queue with weight 1.
- Mixed resource allocation mode (Technical Preview): In this mode, you can specify the resources in mixed types by updating the XML configuration using Cloudera Manager. You can specify the actual units of vcores and memory resources for each queue or specify the percentage of the total resources used by each queue or specify a weight for each queue. You can mix the allocation modes. The queues under one parent can also mix their modes. For more information, see *Setting capacity using mixed resource allocation mode*.

Important: This is a technical preview feature and is considered under development. Do not use this in your production systems. If you have feedback, contact Support by logging a case on our Cloudera Support Portal. Technical preview features are not guaranteed troubleshooting and fixes.

Note: When migrating from a CDH cluster and performing the fs2cs scheduler conversion, the default resource allocation mode is weight mode. For more information see, *Migrating Fair Scheduler to Capacity Scheduler*.

Before you begin

- If you are changing to weight resource allocation mode, delete any managed parent queues.
- If you are changing from weight resource allocation mode, delete any dynamic parent queues.

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service.

   A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the root and select View Edit Queue Properties .

   The Queue Properties dialog box is displayed.
3. Select the resource allocation mode.
4. Click Save.
5. Enter yes in the Switch Allocation Mode dialog box and click OK.

   As a result, Queue Manager calculates and updates the resource allocations for all the existing queues. If required, you can further modify the resource allocation.
6. Click Save.
7. Restart YARN and YARN Queue Manager services.

Related Information

Resource allocation overview
Adding queues using YARN Queue Manager UI
Migrating Fair Scheduler to Capacity Scheduler
Migrating Fair Scheduler to Capacity Scheduler
Setting capacity using mixed resource allocation mode (Technical Preview)

Starting and stopping queues

Queues in YARN can be in two states: RUNNING or STOPPED. A RUNNING state indicates that a queue can accept application submissions, and a STOPPED queue does not accept application submissions. The default state of any configured queue is RUNNING.

In Capacity Scheduler, parent queues and leaf queues can be stopped. For an application to be accepted at any leaf queue, all the queues in the hierarchy all the way up to the root queue must be running. This means that if a parent queue is stopped, all of the descendant queues in that hierarchy are inactive, even if their own state is RUNNING.

To stop a queue:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select Stop Queue .
3. You will be prompted for a confirmation. Click OK to stop the queue.

To start a queue:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select Start Queue .
3. You will be prompted for a confirmation. Click OK to start the queue.

Administrators can use the ability to stop and drain applications in a queue for a number of reasons, such as when decommissioning a queue and migrating its users to other queues. Administrators can stop queues at run-time, so that while current applications run to completion, no new applications are accepted. Existing applications can continue until they finish running, and thus the queue can be drained gracefully without any end-user impact.

Deleting queues

You must first stop the queue before deleting the queue. You can delete a single queue and also parent queue and its children if all the queues in the hierarchy are stopped.

In Capacity Scheduler, parent queues, child queues, and the root queue can all be stopped. For an application to be accepted at any child queue, all the queues in the hierarchy all the way up to the root queue must be running. This means that if a parent queue is stopped, all of the descendant queues in that hierarchy are inactive, even if their own state is RUNNING.



Note: If the queue is associated with one or more partitions, you must first set the partition capacity to zero using Edit Child Queue for that queue for all the partitions before you delete the queue.

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select Delete Queue .

   You can use the Delete Queues and its Children option to delete both parent queue and its children queues.
3. You will be prompted for a confirmation. Click OK to stop the queue.

Setting queue priorities

By setting queue priorities you can ensure that applications can access cluster resources.

About this task

Although the preemption feature, which is enabled by default, improves cluster resources use, there are some use cases where applications might not have access to cluster resources without setting priorities:

Long-running applications:

Without setting queue priorities, long-running applications in queue that are under capacity and with lower relative resource use may not release cluster resources until they finish running.

Applications that require large containers:

The issue with long-running applications is exacerbated for applications that require large containers. With short-running applications, previous containers may eventually finish running and free the cluster resources for applications with large containers. But with long-running services in the cluster, the large containers may never get sufficiently large resources on any nodes.

Hive LLAP

Hive LLAP (Low-Latency Analytical Processing) enables you to run Hive queries with low-latency in near real-time. To ensure low-latency, you should set the priority of the queue used for LLAP to a higher priority, especially if your cluster includes long-running applications.

For more information, see *Configuring preemption* and *Enabling Intra-Queue preemption*

Before you begin

Configure preemption as described in *Configuring preemption*.

Procedure

1. In Cloudera Manager navigate to Cluster YARN Queue Manager UI .
   The Overview tab is opened, displaying a hierarchical view of the queues.
2. Find the queue for which you want to set queue priority and click the three vertical dots next to its name.
3. Select Edit Queue Properties.
4. In the Resource Allocation section find the Queue Priority property.
5. Set the queue priority.
   By default all queues are set to a priority of 0. Higher numbers indicate higher priority.
6. Click Save.

Related Information

Configuring preemption
Enabling Intra-Queue preemption

Configuring scheduler properties at the global level

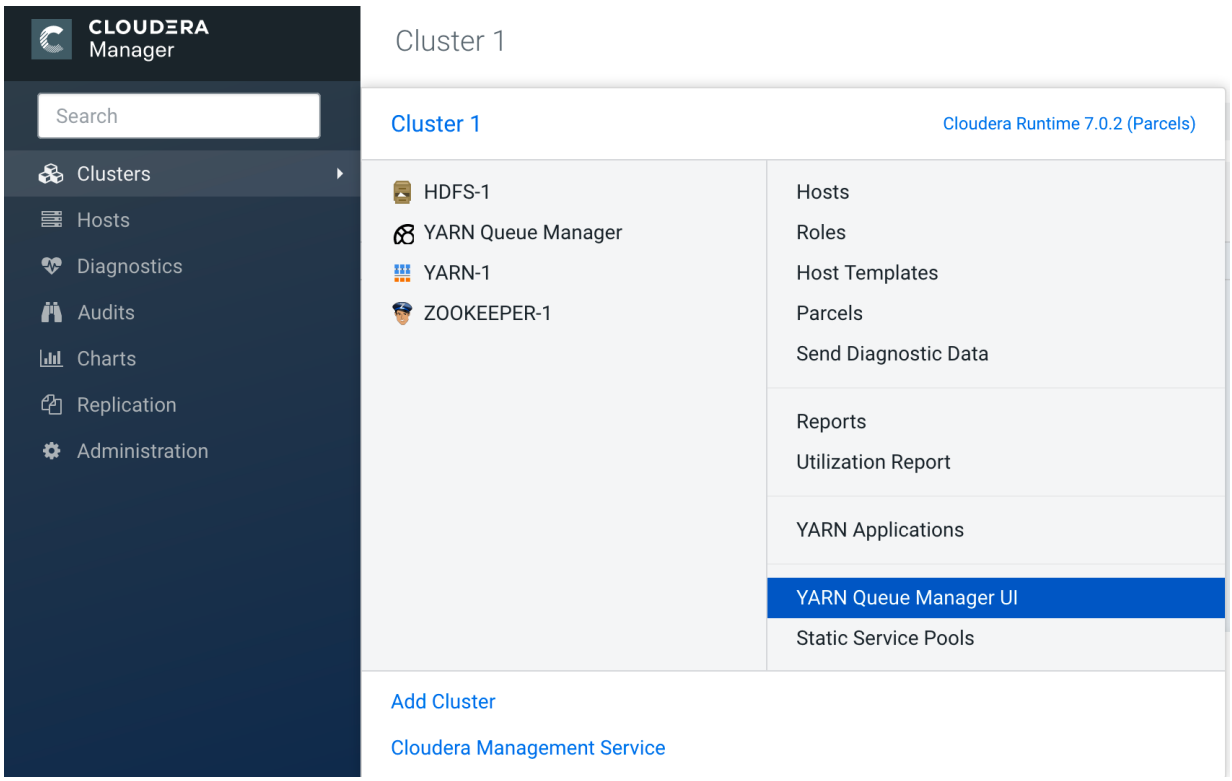
You can configure the scheduler properties to define the behaviour of all the queues. All parent and children queues inherit the properties set using the Scheduler Properties.

About this task

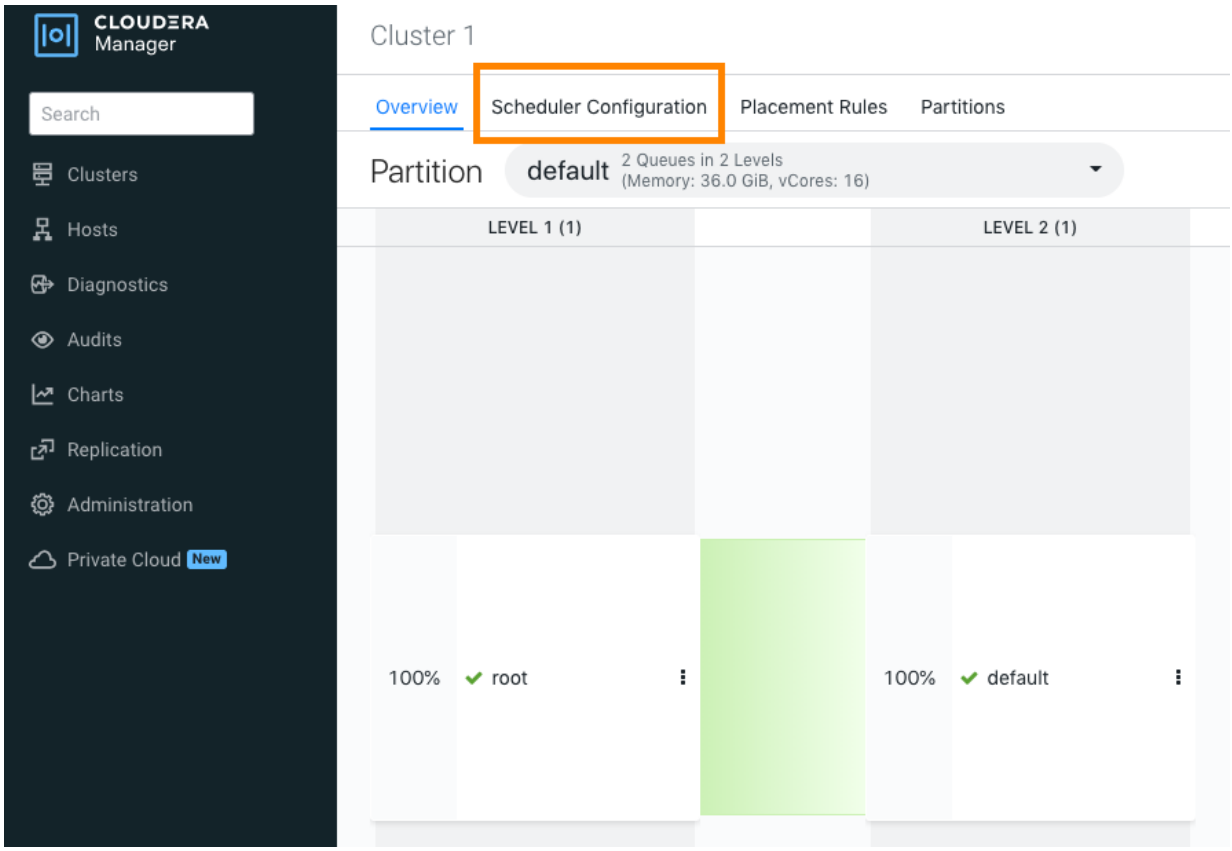
In Cloudera Manager, you can use the Configuration tab to configure the scheduler properties.

Procedure

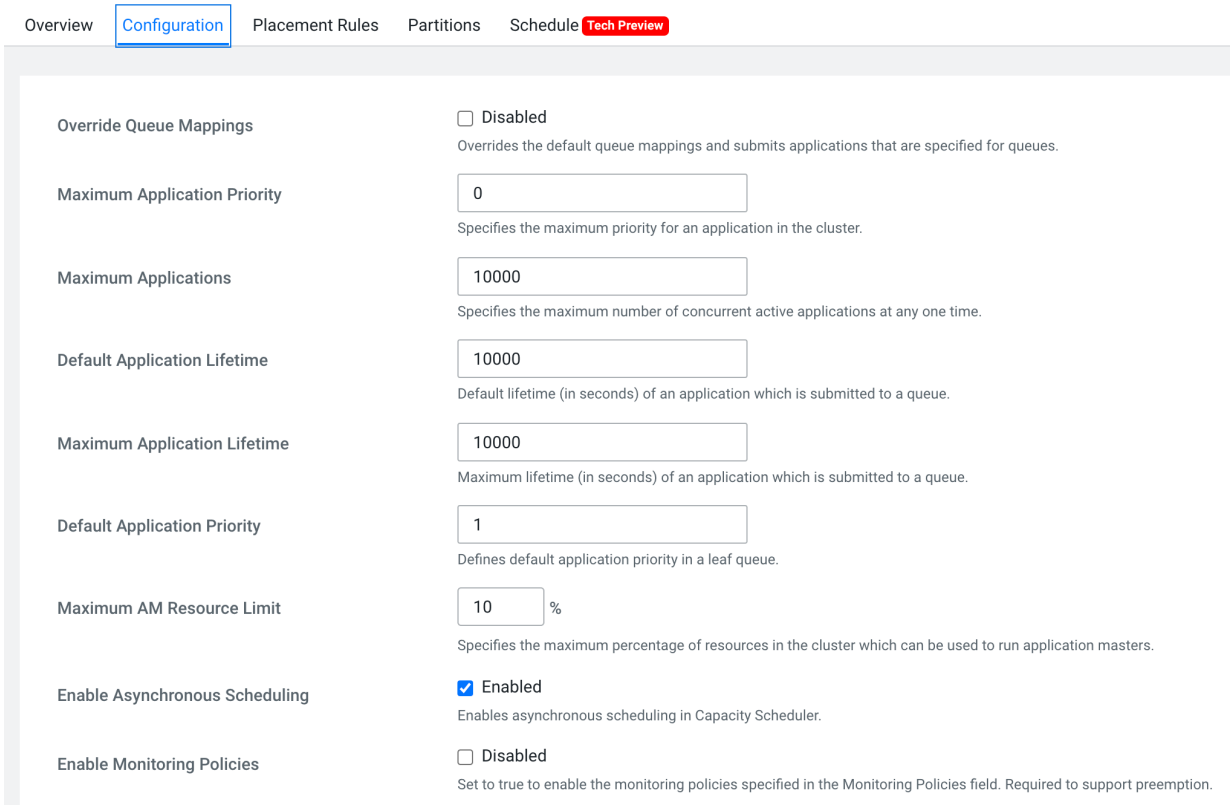
1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service.



2. In the YARN Queue Manager window, click on the Configuration tab.



3. In the Scheduler Configuration window, enter the value of the property and click Save.



Setting global maximum application priority

You can use Priority Scheduling to run YARN applications at higher priority, regardless of other applications that are already running in the cluster. YARN allocates more resources to applications running at a higher priority over those running at a lower priority. Priority Scheduling enables you to set an application's priority both at the time of submission and dynamically at run time.

About this task

Priority Scheduling works only with the FIFO (First In, First Out) ordering policy. FIFO is the default Capacity Scheduler ordering policy.

To set appliation priority (yarn.cluster.max-application-priority), perform the following steps:

Procedure

1. In Cloudera Manager, select the YARN service.
2. Click Configuration.
3. Search for yarn-site.xml.
4. Under YARN Service Advanced Configuration Snippet (Safety Valve) for yarn-site.xml, add the yarn.cluster.max-application-priority parameter and the value you need.
5. Click Save Changes.
6. Restart the YARN service.

Configuring preemption

Preemption allows applications of higher priority to preempt applications of lower priority.

About this task

A scenario can occur where a queue has a guaranteed level of cluster resources, but must wait to run applications because other queues are utilizing all of the available resources. If preemption is enabled for a particular queue, applications of higher priority do not have to wait because applications of lower priority have taken up the available capacity. When you enable Preemption (yarn.resourcemanager.scheduler.monitor.enable) the allocated cluster resources are taken from the preemption enabled queue utilizing more than their guaranteed capacity and served to the underserved queues immediately, without having to wait for other queues' applications to finish running.

You can use Priority Scheduling to run YARN applications at a higher priority, regardless of other applications that are already running in the cluster . For more information, see Setting global maximum application priority on page 39.

To configure the Queue Preemption options on all queues, perform the following:

Procedure

1. In Cloudera Manager, navigate to YARN Configuration .
2. Search for preemption.
3. Find the Capacity Scheduler Preemption property.
4. Ensure that it is checked, meaning that Preemption is enabled.
5. Under Configuration, search for ResourceManager Advanced Configuration Snippet (Safety Valve) for yarn-site.xml.
6. Configure the required preemption properties:
   - Preemption: Observe Only - Add the yarn.resourcemanager.monitor.capacity.preemption.observe_only parameter and the value you need.

     It runs the policy, but does not affect the cluster with preemption and stop events.
   - Preemption: Monitoring Interval (ms) - Add the yarn.resourcemanager.monitor.capacity.preemption.monitoring_interval parameter and the value you need.

     The time in milliseconds between invocations of this policy. Setting this value to a longer time interval causes the Capacity Monitor to run less frequently.
   - Preemption: Maximum Wait Before Kill (ms) - Add the yarn.resourcemanager.monitor.capacity.preemption.max_wait_before_kill parameter and the value you need.

     The time in milliseconds between requesting a preemption from an application and stopping the container. Setting this to a higher value gives applications more time to respond to preemption requests and gracefully release containers.
   - Preemption: Total Resources Per Round - Add the yarn.resourcemanager.monitor.capacity.preemption.total_preemption_per_round parameter and the value you need.

     The maximum percentage of resources preempted in a single round. You can use this value to restrict the pace at which containers are reclaimed from the cluster. After computing the total desired preemption, the policy scales it back to this limit.
   - Preemption: Over Capacity Tolerance - Add the yarn.resourcemanager.monitor.capacity.preemption.max_ignored_over_capacity parameter and the value you need.

     The maximum amount of resources above the target capacity ignored for preemption. The default value is 0.1, which means that the Resource Manager starts preemption for a queue only when it goes 10% above its guaranteed capacity. This avoids resource-rotation and aggressive preemption.
   - Preemption: Maximum Termination Factor - Add the yarn.resourcemanager.monitor.capacity.preemption.natural_termination_factor parameter and the value you need.

The maximum percentage of each queue's preemption target capacity that is preempted per cycle. You can increase this value to speed up resource reclamation.

7. Click Save Changes.
8. Restart the YARN services.

   For information about configuring preemption for a specific queue, see Configure preemption for a specific queue.

Enabling Intra-Queue preemption

Intra-queue preemption prevents resource imbalances in a queue.

Before you begin

Enable preemption by setting the yarn.resourcemanager.scheduler.monitor.enable property to true.

About this task

Intra-queue preemption helps in effective distribution of resources within a queue based on configured user limits or application priorities.

To configure Intra-Queue Preemption (yarn.resourcemanager.monitor.capacity.preemption.intra-queue-preemption.enabled) on all queues, perform the following steps:

Procedure

1. In Cloudera Manager, select the YARN service.
2. Click Configuration.
3. Search for yarn-site.xml.
4. Under YARN Service Advanced Configuration Snippet (Safety Valve) for yarn-site.xml, add the yarn.resourcemanager.monitor.capacity.preemption.intra-queue-preemption.enabled parameter and the value to true.
5. Click Save Changes.
6. Restart the YARN service.

   For information about configuring intra-queue preemption for a specific queue, see Configure Intra-Queue Preemption for a specific queue.

Enabling LazyPreemption

By enabling the LazyPreemption feature you can optimize the selection of containers from queues for preemption. Containers are preempted only if the resources can be used by another application.

About this task

Normally when preemption is performed there is no guarantee that preempted resources can be used by pending resource requests from under-satisfied queues. That can lead to excessive preemption as the preemption monitor keeps preempting resources from queue-A, but queue-B cannot use the preempted resources. As a result, such sources are allocated to queue-A again, but then they are preempted again.

The LazyPreemption feature makes resource preemption more efficient. It prevents excessive preemption and ensures that container selection is continuously optimized. It is diabled by default.

Enabling the LazyPreemption feature has the following advantages:

- Containers are killed only if the demanding application can use that capacity on the specific nodes.
- Application requests that cannot be satisfied at a moment for some reason will not lead to any container kill.

- If the application request gets canceled, no containers are killed.

LazyPreemption avoids scenarios when preempted resources cannot be used by the application that needs the resource, and then such preempted resources go back to the over-utilized queue again.

Procedure

1. In Cloudera Manager, navigate to YARN Configuration .
2. Search for safety valves.
3. Find the Capacity Scheduler Configuration Advanced Configuration Snippet (Safety Valve) property.
4. Set the yarn.scheduler.capacity.lazy-preemption-enabled property to true by adding the following code snipper:

```
<property>
   <name>yarn.scheduler.capacity.lazy-preemption-enabled</name>
   <value>true</value>
 </property>
 <property>
```

5. Click Save Changes.

Results

LazyPreemption is enabled.

Setting global application limits

To avoid system-thrash due to an unmanageable load -- caused either by malicious users, or by accident -- the Capacity Scheduler enables you to place a static, configurable limit on the total number of concurrently active (both running and pending) applications at any one time.

You can set the maximum applications limit (yarn.scheduler.capacity.maximum-applications) using this configuration property. The default value is 10,000.

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the Scheduler Configuration tab.
3. Enter the maximum application limit in the Maximum Applications text box.
4. Click Save.

For information about overriding this limit on a per-queue basis, see Set Application limit for a specific queue.

Related Information

Setting maximum parallel application limits

Setting maximum parallel application limits for a specific queue

Setting default Application Master resource limit

The Application Master (AM) resource limit that can be used to set a maximum percentage of cluster resources allocated specifically to Application Masters. This property has a default value of 10%, and exists to avoid cross-application deadlocks where significant resources in the cluster are occupied entirely by the Containers running ApplicationMasters.

About this task

The Maximum AM Resource Limit (yarn.scheduler.capacity.maximum-am-resource-percent) property also indirectly controls the number of concurrent running applications in the cluster, with each queue limited to a number of running applications proportional to its capacity.

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Configuration tab.
3. Enter the maximum AM resource limit in the Maximum AM Resource Limit text box.
4. Click Save.

   For information about overriding this limit on a per-queue basis, see Configure Application Master resource limit for a specific queue.

Related Information

Setting maximum parallel application limits

Setting maximum parallel application limits for a specific queue

Enabling asynchronous scheduler

Asynchronous scheduler decouples the CapacityScheduler scheduling from Node Heartbeats. This improves the latency significantly.

About this task

To enables asynchronous scheduling (yarn.scheduler.capacity.schedule-asynchronously.enable), perform the following:

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Configuration tab.
3. Select the Enable Asynchronous Scheduler check-box.
4. Click Save.

Configuring queue mapping to use the user name from the application tag using Cloudera Manager

You learn how to add service users to the YARN queue by following a mapping procedure.

You can configure queue mapping to use the user name from the application tag instead of the proxy user who submitted the job. You can add only service users like hive using the `yarn.resourcemanager.application-tag-based-placement.username.whitelist` property and not normal users.

When a user runs Hive queries, HiveServer2 submits the query in the queue mapped from an end user instead of a hive user. For example, when user *ALICE* submits a Hive query with doAs=false mode, job will run in YARN as hive user. If application-tag based scheduling is enabled, then the job will be placed to a target queue based on the queue mapping configuration of user *ALICE*.

For more information about queue mapping configurations, see Manage placement rulesManage placement rules. For information about Hive access, see Apache Hive documentation.

How to configure queue mapping

1. In Cloudera Manager, select the YARN service.
2. Click the Configuration tab.
3. Search for ResourceManager. In the Filters pane, under Scope, select ResourceManager.

4. In ResourceManager Advanced Configuration Snippet (Safety Valve) for yarn-site.xml add the following:

   a. Enable the application-tag-based-placement property to enable application placement based on the user ID passed using the application tags.

   ```
   Name: yarn.resourcemanager.application-tag-based-placement.enable
   Value: true
   Description: Set to "true" to enable application placement based on the
    user ID passed using the application tags. When it is enabled, it chec
   ks for the userid=<userId> pattern and if found, the application will be
    placed onto the found user's queue, if the original user has the requir
   ed rights on the passed user's queue.
   ```

   b. Add the list of users in the allowlist who can use application tag based placement. The applications when the submitting user is included in the allowlist, will be placed onto the queue defined in the yarn.scheduler.capacity.queue-mappings property defined for the user from the application tag. If there is no user defined, the submitting user will be used.

   ```
   Name: yarn.resourcemanager.application-tag-based-placement.username.whit
   elist
   Value: <Hive process user(s)>
   Description: Comma separated list of users who can use the application t
   ag based placement, if "yarn.resourcemanager.application-tag-based-place
   ment.enable" is enabled.
   ```

   Note: Check the Hive system user value(s) and add the value(s) to the allowlist:

   1. In Cloudera Manager, navigate to Hive Configuration .
   2. Search for System User.
   3. Note down the value(s) set as process username(s), and add the value(s) to the allowlist.
   4. When there are no rules mapping for the tag user, it will fallback to the proxy user's (Hive) placement rule.

5. Restart the ResourceManager service for the changes to apply.

Related Information

Managing placement rules

Configuring NodeManager heartbeat

You can control how many containers can be allocated in each NodeManager heartbeat.

To configure NodeManager heartbeat for all queues, perform the following:

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Scheduler Configuration tab.
3. Select the Enable Multiple Assignments Per Heartbeat checkbox to allow multiple container assignments in one NodeManager heartbeat.
4. Configure the following NodeManager heartbeat properties:
   - Maximum Container Assignments Per Heartbeat - The maximum number of containers that can be assigned in one NodeManager heartbeat. Setting this value to -1 disables this limit.
   - Maximum Off-Switch Assignments Per Heartbeat - The maximum number of off-switch containers that can be assigned in one NodeManager heartbeat.
5. Click Save.

Configuring data locality

Capacity Scheduler leverages Delay Scheduling to honor task locality constraints. There are three levels of locality constraint: node-local, rack-local, and off-switch. The scheduler counts the number of missed opportunities when the locality cannot be satisfied and waits for this count to reach a threshold before relaxing the locality constraint to the next level. You can configure this threshold using the Node Locality Delay (yarn.scheduler.capacity.node-locality-delay) and Rack Locality Additional Delay (yarn.scheduler.capacity.rack-locality-additional-delay) fields.

To configure data locality, perform the following:

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Scheduler Configuration tab.
3. In the Node Locality Delay textbox, enter the number of scheduling opportunities that can be missed.

   Capacity Scheduler attempts to schedule rack-local containers only after missing this number of opportunities. You must ensure this number is same as the number of nodes in the cluster.
4. In the Rack Locality Additional Delay textbox, enter the number of missed scheduling opportunities, over the Node Locality Delay ones, after which Capacity Scheduler should attempt to schedule off-switch containers.

   A value of -1 means that the value is calculated based on the formula L * C / N, where L is the number of locations (nodes or racks) specified in the resource request, C is the number of requested containers, and N is the size of the cluster.
5. Click Save.

Setting Maximum Parallel Application

You can set the maximum number of applications that can run at the same time in a cluster. Ensure you understand how the parallel application limits are inherited from the parent queue in the queue hierarchy before using this option.

You can set parallel application limits for all queues, all users, and at the user level. When the maximum parallel applications limit is reached, application submissions are not rejected and instead they stay in the *ACCEPTED* state until they are eligible to run.

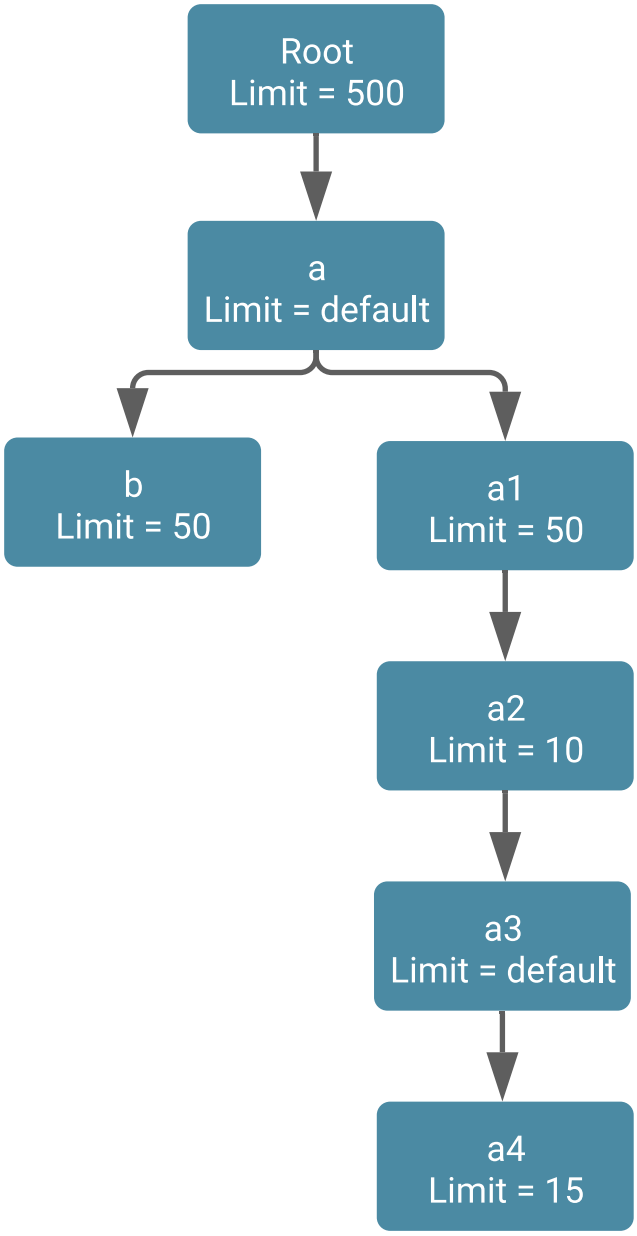
The evaluation of the limits happens in the following order:

1. Maximum Applications - if the limit is exceeded, the submission is rejected immediately.
2. Maximum Parallel Applications - if the limit is exceeded, the submission is accepted, but the application does not transition to the RUNNING state. It stays in the ACCEPTED state until the queue or user limits are satisfied.
3. Maximum AM Resource Limit - if there are too many Application Masters running, the application stays in the ACCEPTED state until the resources become available.

Example - Application Limits

The maximum parallel application limit is an inherited property in the queue hierarchy. You can lower the limit down the hierarchy, but setting a higher value than its higher queue in the queue structure, the limit will not be effective. The following example provides information about how the parallel application limit is applied in a queue hierarchy. The Maximum Parallel Applications: All Queues value is set to 250 at the scheduler level and 500 is set at the root level.

Example queue hierarchy:



- Maximum 250 applications can run parallel under the root.a queues.
- Maximum 50 applications can run parallel under the root.a.a1 queues.
- Maximum 10 applications can run parallel under the root.a.a1.a2 queues.
- Maximum 10 applications can run parallel under the root.a1.a2.a3 queues. Even though the Maximum Parallel Applications limit is not set for .a3, the default value of 250 applies for that queue; however, the parent queue (root.a.a1.a2) has a lower value of 10, and the child .a3 queue cannot exceed that limit.
- Maximum 10 applications can run parallel under the root.a1.a2.a3.a4 queue. Even though the queue level limit is configured for 15 applications, the parents restrict this limit to 10.
- Maximum 50 applications can run parallel under the root.a.b queue.

Example - User Limits

- Scenario 1 - In this scenario, there are two users, *USER1* and *USER2*. When you set the Maximum Parallel Applications: All Queues to 100 and Maximum Parallel Applications: All Users to 50, both *USER1* and *USER2* can have 50 applications running in parallel.

- Scenario 2 - In this scenario, there are four users, *USER1*, *USER2*, *USER3*, and *USER4*. You can set the limits as follows:
  - Maximum Parallel Applications: All Queues - 100
  - Maximum Parallel Applications: All Users - 15
  - Maximum Parallel Applications: Per Users
    - User1 - 10
    - User 2 - 25

  USER1 can have 10 applications running in parallel. *USER2* can have 25 applications running in parallel. *USER3* and *USER4* can have 15 applications running in parallel.
- Scenario 3: In this scenario, the Maximum Parallel Applications limit for root.a is set to 5. The Maximum Parallel Applications: Per Users for *USER1* is set to 10. Even though *USER1* can run 10 parallel applications, in the root.a queue, *USER1* can run only 5 maximum applications. *USER1* can also run 5 more applications on some other queue.
- Scenario 4: In this scenario, the Maximum Parallel Applications limit for root.b is set to 20. The Maximum Parallel Applications: Per Users limit for *USER2* is set to 10. *USER2* can run 10 parallel applications, in the root.b queue. Some other users can run 10 more applications in the root.b queue.

Related Information

Setting maximum parallel application limits

Setting maximum parallel application limits

You can set the maximum number of applications limits for all queues, all users, and at the user level. The maximum parallel application limit is inherited from the “root” queue level and is lowered down in the queue hierarchy. The limit is checked in the queue hierarchy and the lowest value is applied as the limit.

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Configuration tab.
3. Specify the application limits:
   - Maximum Parallel Applications: All Queues - specifies the maximum number of applications that can run at the same time for any queue that does not override it with the per queue level setting.
   - Maximum Parallel Applications: All Users - specifies the maximum number of applications that can run at the same time for all the users. This limit is overridden based on the per user limit.
   - Maximum Parallel Applications: Per User - specifies parallel running application limit for the specified user. Click + to add more users and their application limits.
4. Click Save.

Related Information

Setting Maximum Parallel Application
Setting maximum parallel application limits for a specific queue
Setting default Application Master resource limit
Setting global application limits

Configuring per queue properties

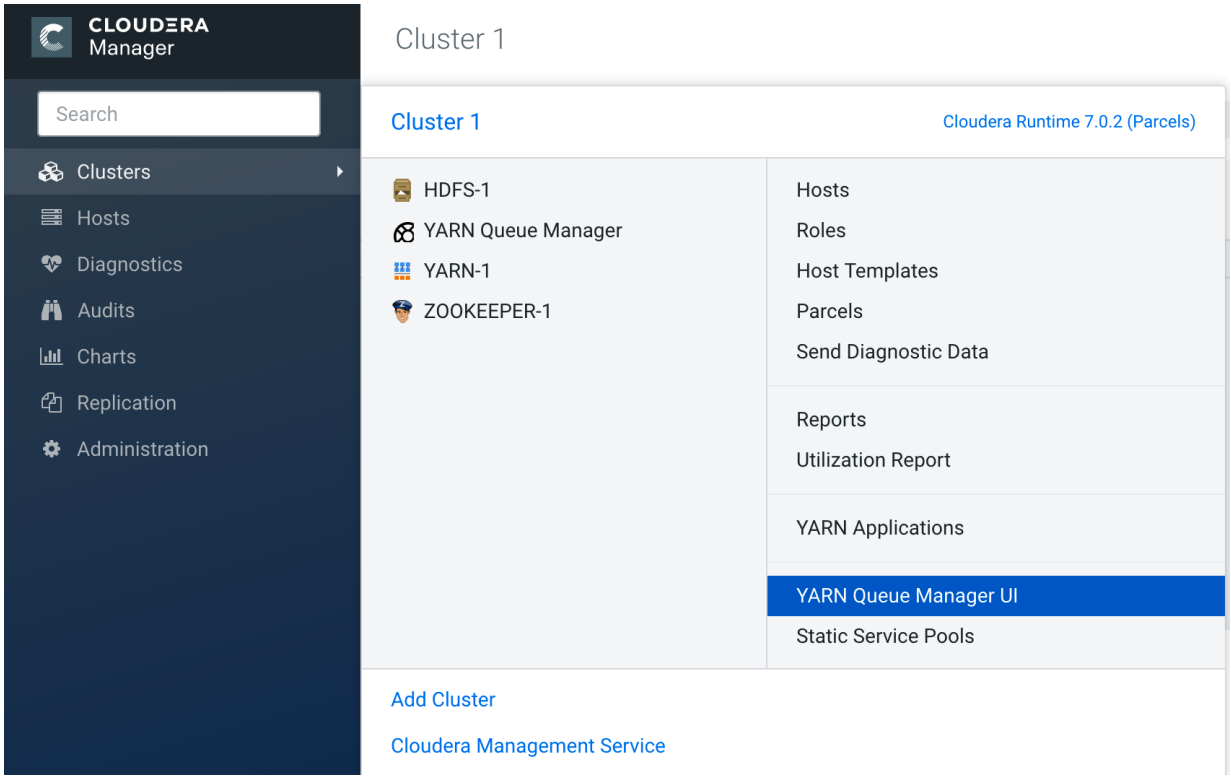
Queue properties contain the settings that define the behavior of the queue. Using queue properties, you can define the setting that need not inherit properties directly from the parent queue and define the setting specific to the queue.

About this task

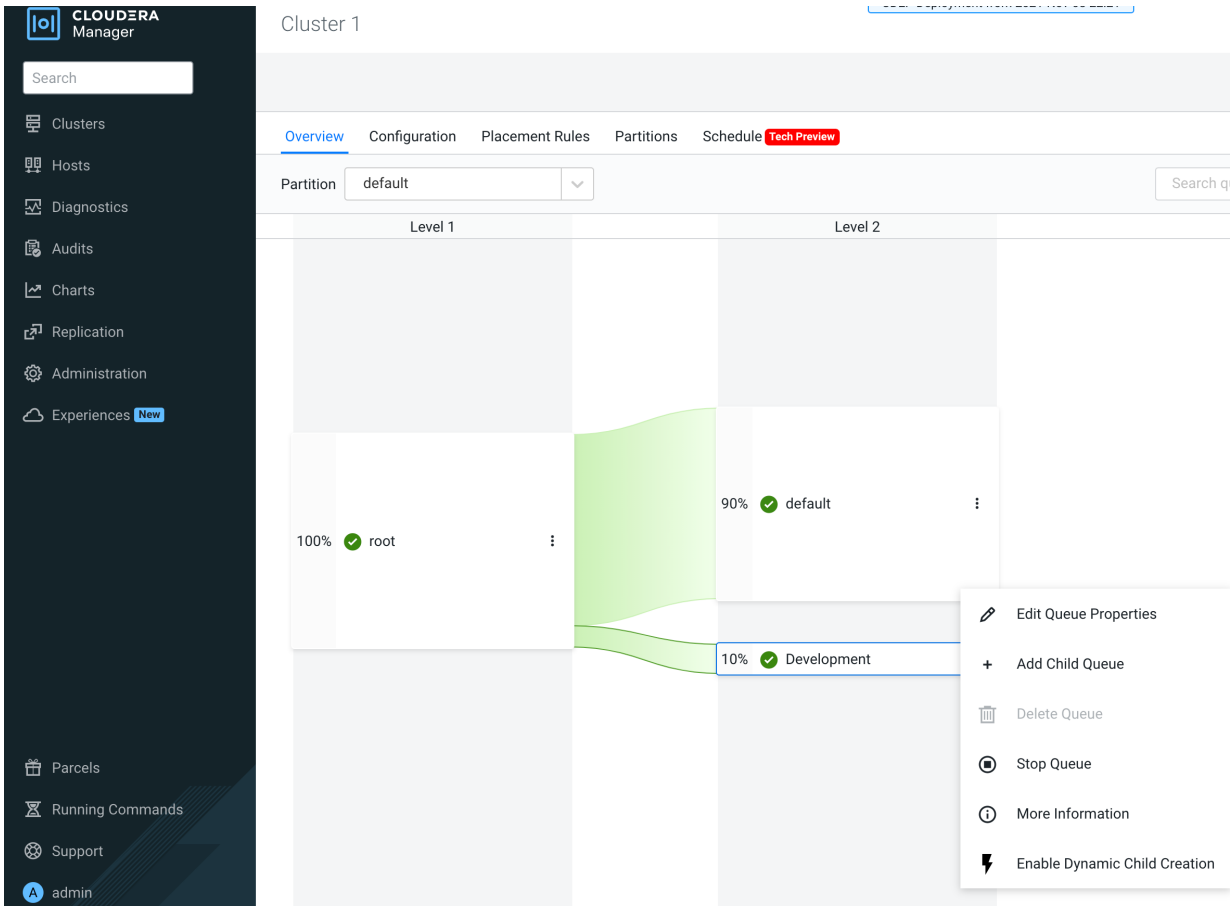
In Cloudera Manager, you can use the Queue Properties to view and configure the queue properties.

Procedure

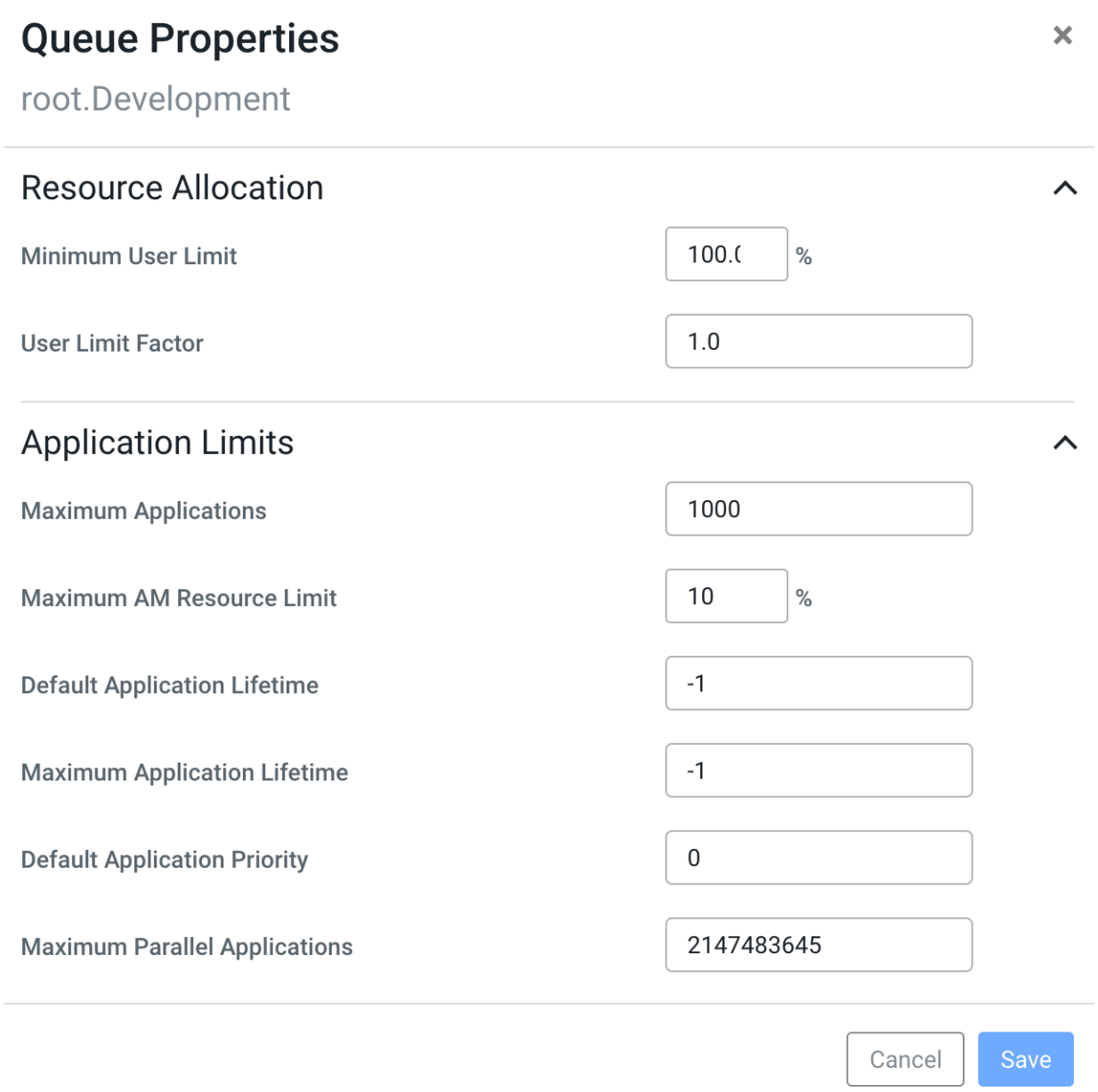
1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service.



2. Click on the three vertical dots on the queue and select the Edit Queue Properties option.



3. In the Queue Properties window, enter the value of the property and click Save.



Setting user limits within a queue

Set a minimum percentage of resources allocated to each leaf queue user.

Minimum User Limit

The Minimum User Limit (minimum-user-limit-percent) property can be used to set the minimum percentage of resources allocated to each leaf queue user. The Minimum User Percentage is a soft limit on the smallest amount of resources a single user should get access to if they are requesting it. For example, if you set the Minimum User Percentage property to 10%, 10 users get 10% each, assuming they are all requesting it; this value is a soft limit because if one of the users asks for less capacity, you can place more users in the queue.

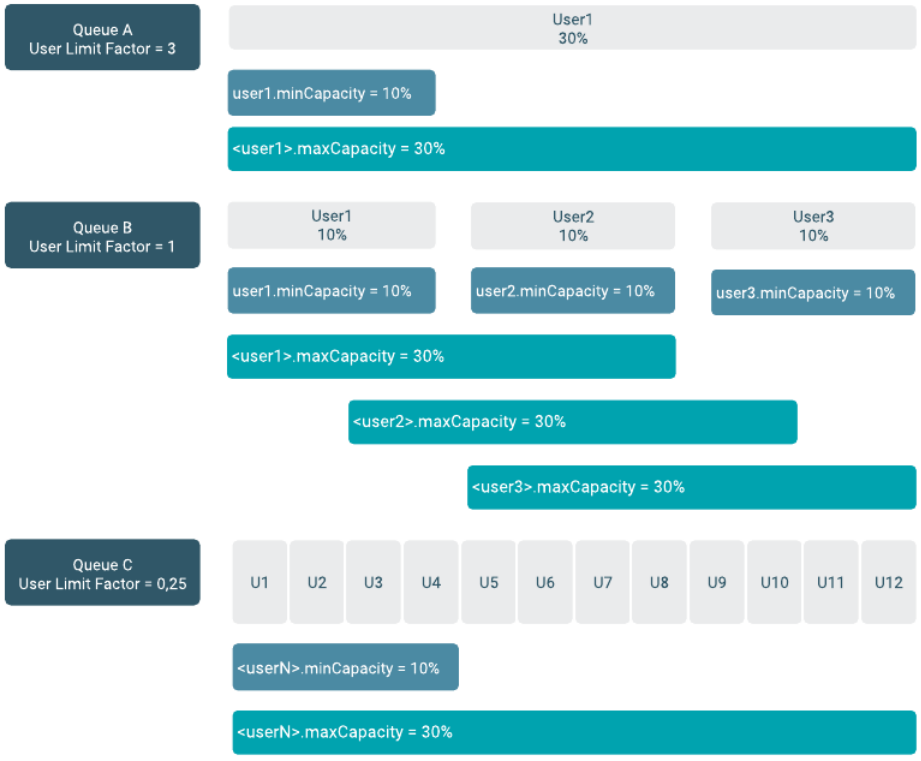
For example, to enable equal sharing of the *SERVICES* leaf queue capacity among five users, you must set the Minimum User Limit value to 20%.

To configure user limits based on this example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the Services queue and select Edit Queue Properties option.
3. In the Queue Properties dialog-box, enter 20 in the Minimum User Limit text box.
4. Click Save.

User Limit Factor

You can use the User Limit Factor (user-limit-factor) to control the maximum amount of resources that a single user can consume. User Limit Factor is set as a multiple of the queues minimum capacity where a user limit factor of one means the user can consume the entire minimum capacity of the queue. If the user limit factor is greater than 1, it is possible for the user consumption to grow into maximum capacity and if the value is set to less than 1, such as 0.5, a user can only obtain half of the minimum capacity of the queue. If you want a user to obtain the maximum capacity of a queue, set a value greater than 1 to allow the minimum capacity to be overtaken by a user that many times.



The following table shows how the queue resources are adjusted as users submit jobs to a queue with a minimum user limit percentage is set to 20%:

Table 1: Minimum User Limit

Minimum user limit percent = 20	
1st user submits job	Sole user gets 100% of the queue capacity
2nd user submits job	Each user equally shares 50% of the queue capacity
3rd user submits job	Each user equally shares 30.33% of the queue capacity
4th user submits job	Each user equally shares 25% of the queue capacity
5th user submits job	Each user equally shares 20% of the queue capacity
6th user submits job	6th user must wait for queue capacity to free up

- Queue resources are adjusted in the same manner for a single user submitting multiple jobs in succession. If no other users are requesting queue resources, the first job receives 100% of the queue capacity. When the user submits a second job, each job receives 50% of queue capacity. When the user submits a third job, each job receives 33% of queue capacity. If a fourth user then submits a job, each job would receive 25% of queue capacity. When the number of jobs submitted by all users reaches a total of five, each job will receive 20% of queue capacity, and subsequent users must wait for queue capacity to free up (assuming preemption is not enabled).
- The Capacity Scheduler also manages resources for decreasing numbers of users. As users' applications finish running, other existing users with outstanding requirements begin to reclaim that share.
- Note that despite this sharing among users, the FIFO application scheduling order of Capacity Scheduler does not change. This guarantees that users cannot monopolize queues by submitting new applications continuously. Applications (and thus the corresponding users) that are submitted first always get a higher priority than applications that are submitted later.

Capacity Scheduler's leaf queues can also use the user-limit-factor property to control user resource allocations. This property denotes the fraction of queue capacity that any single user can consume up to a maximum value, regardless of whether or not there are idle resources in the cluster.

To configure the maximum limit (user-limit-factor) based on this example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue you want to set limit and select Edit Queue Properties option.
3. In the Queue Properties dialog-box, enter 1 in the User Limit Factor text box.
4. Click Save.

The default value of 1 means that any single user in the queue can, at the most, only occupy only the queue's configured capacity. This prevents users in a single queue from monopolizing resources across all queues in a cluster. Setting the value to 2 restricts the queue's users to twice the queue's configured capacity. Setting it to a value of 0.5 restricts any user from using resources beyond half of the queue capacity.

Setting Maximum Application limit for a specific queue

You can set the maximum number of applications that can run in a specific queue. This limit overrides it's parent queue limit if this specific queue level value is lesser than what is set for the parent queue.

About this task

You can set the maximum-application-limit property using the Maximum Application queue property. The limit for running applications in any specific queue is a fraction of this total limit, proportional to its capacity. This is a hard limit, which means that once this limit is reached for a queue, any new applications to that queue will be rejected, and clients will have to wait and retry later.

To set the application limit (yarn.scheduler.capacity.<queue-path>.maximum-applications) on a specific queue, perform the following:

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select View/Edit Queue Properties option.
3. In the Queue Properties dialog-box, enter the maximum application limit in the Maximum Application text box.
4. Click Save.

   For information about setting application limits on all the queues, see Set Application Limits.

Setting Application-Master resource-limit for a specific queue

The Application Master (AM) resource limit can be used to set a maximum percentage of cluster resources allocated specifically to Application Masters. The default value is 10% and exists to avoid cross-application deadlocks where significant resources in the cluster are occupied entirely by the Containers running Application Masters.

About this task

This property also indirectly controls the number of concurrent running applications in the cluster, with each queue limited to a number of running applications proportional to its capacity.

To set the maximum Application Masters resource limit (yarn.scheduler.capacity.maximum-am-resource-percent) for a specific queue:

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select View/Edit Queue Properties option.
3. In the Queue Properties dialog-box, enter the limit in the Maximum AM Resource Limit text box.
4. Click Save.

   For information about setting Application Master resource limits on all the queues, see Set Application Master Resource Limit.

Setting maximum parallel application limits for a specific queue

You can set the maximum number of applications that can run in a specific queue. This limit overrides it's parent queue limit if this specific queue level value is lesser than what is set for the parent queue.

Before you begin

Learn how the parallel application limits are inherited from the parent queue in the queue hierarchy and its evaluation. For more information, see Maximum Parallel Application settings .

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select Edit Queue Properties option.
3. In the Queue Properties dialog-box, enter the maximum parallel application limit in the Maximum Parallel Applications text box.
4. Click Save.

Related Information

Setting maximum parallel application limits

Setting default Application Master resource limit

Setting global application limits

Controlling access to queues using ACLs

Use Access-control lists (ACLs) to control access rights to users and administrators to the Capacity Scheduler queues.

Application submission can really only happen at the leaf queue level, but an ACL restriction set on a parent queue will be applied to all of its descendant queues.

In the Capacity Scheduler, ACLs are configured by granting queue access to a list of users and groups with the Submit Application ACL parameter. The format of the list is "user1,user2 group1,group2" -- a comma-separated list of users, followed by a space, followed by a comma-separated list of groups.



Note: The default value of Submit Application ACL for a root queue is yarn, which means that only the default yarn user can submit applications to that queue. Therefore, to provide specific users and groups with access to the queue, you must explicitly set the value of Submit Application ACL to those users and groups.

The Submit Application ACL (acl_submit_applications) parameter can also be set to the following values:

asterisk ("*")

Allow access to all users and group

space character (" ")

Block access to all users and groups

empty string value ("")

Block access to all users and groups

As mentioned previously, ACL settings on a parent queue are applied to all of its descendant queues. Therefore, if the parent queue uses the "*" (asterisk) value (or is not specified) to allow access to all users and groups, its child queues cannot restrict access. Similarly, before you can restrict access to a child queue, you must first set the parent queue to " " (space character) or "" (empty string value) to block access to all users and groups.

For example, the following properties would set the root Submit Application ACL value to " " (space character) to block access to all users and groups, and also restrict access to its child "support" queue to the users "sherlock" and "john" and the members of the "cfo-group" group:

Each child queue is tied to its parent queue with the configuration property. The top-level "support", "engineering", and "marketing" queues would be tied to the "root" queue.

To set the ACLs based on this example, perform the following:

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue you want to set ACL and select Edit Queue Properties option.
3. In the Queue Properties dialog-box, add sherlock,john cfo-group in the Submit Application ACL text box.
4. Click Save.

A separate ACL can be used to control the administration of queues at various levels. Queue administrators can submit applications to the queue, stop applications in the queue, and obtain information about any application in the queue (whereas normal users are restricted from viewing all of the details of other users' applications).

If the Queue Administer ACL value is set to " " (space character) or "" (empty string value), it blocks access to all users and groups. If the ACL is set to sherlock,john cfo-group , it allows access to the users "sherlock" and "john" and the members of the "cfo-group" group.

Enabling preemption for a specific queue

Capacity Scheduler Preemption allows higher-priority applications to preempt lower-priority applications.

About this task

A scenario can occur in which a queue has a guaranteed level of cluster resources, but must wait to run applications because other queues are utilizing all of the available resources. If Preemption is enabled, higher-priority applications do not have to wait because lower priority applications have taken up the available capacity. With Preemption enabled, the allocated cluster resources are taken from the preemption enabled queue utilizing more than their guaranteed capacity and served to the underserved non-preemption enabled queues immediately, without having to wait for other queues' applications to finish running.

Note: If the preemption policy is disabled in the scheduler configurations, you cannot enable preemption for a specific queue. For information about setting scheduler level preemption, see Configuring preemption on page 39.

You can disable the queue preemption (yarn.resourcemanager.scheduler.monitor.enable) for a specific queue.

Before you begin

To enable or disable preemption at the queue level in the Queue Manager UI, you must enable monitoring policies in the global configuration level and define the monitoring policies' class.

- In the YARN Queue Manager WebUI Configuration , check Enable Monitoring Policies and set Monitoring Policies to org.apache.hadoop.yarn.server.resourcemanager.monitor.capacity.ProportionalCapacityPreemptionPolicy and save to enable/disable preemption for specific queues.

With Enable Monitoring Policies disabled the queue-level preemption option is greyed out and indicates the inherited configuration (enabled by default).

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select View/Edit Queue Properties option.
3. In the Queue Properties dialog-box, uncheck the Enable Preemption checkbox.
4. Click Save.

Enabling Intra-Queue Preemption for a specific queue

Intra-queue preemption prevents resource imbalances in a queue.

About this task

Intra-queue preemption helps in effective distribution of resources within a queue based on configured user limits or application priorities.

Note: If the intra-queue preemption policy is disabled in the scheduler configurations, you cannot enable intra-queue preemption for a specific queue. For information about setting scheduler level intra-queue preemption, see Configure Intra-Queue Preemption.

To disable Intra-Queue Preemption (yarn.resourcemanager.monitor.capacity.preemption.intra-queue-preemption.enabled) for a specific queue.

Before you begin

You must enable inter-queue preemption to enable intra-queue preemption at the queue level.

Procedure

1. In Cloudera Manager select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select View/Edit Queue Properties option.
3. In the Queue Properties dialog-box, uncheck the Enable Intra Queue Preemption checkbox.
4. Click Save.

Setting ordering policies within a specific queue

Set FIFO (First In, First Out) or Fair scheduling policies in Capacity Scheduler depending on your requirements.

The default ordering policy in Capacity Scheduler is FIFO (First In, First Out). FIFO generally works well for predictable, recurring batch jobs, but sometimes not as well for on-demand or exploratory workloads. For these types of jobs, Fair Scheduling is often a better choice. Flexible scheduling policies enable you to assign FIFO or Fair ordering policies for different types of workloads on a per-queue basis.

Examples FIFO and Fair Sharing Policies

Both FIFO (First In, First Out) and Fair scheduling policies work differently in batch jobs and ad hoc jobs.

Batch Example

In the below example, two queues have the same resources available. One uses the FIFO ordering policy, and the other uses the Fair Sharing policy. A user submits three jobs to each queue one after another, waiting just long enough for each job to start. The first job uses 6x the resource limit in the queue, the second 4x, and last 2x.

- In the FIFO queue, the 6x job would start and run to completion, then the 4x job would start and run to completion, and then the 2x job. They would start and finish in the order 6x, 4x, 2x.
- In the Fair queue, the 6x job would start, then the 4x job, and then the 2x job. All three would run concurrently, with each using 1/3 of the available application resources. They would typically finish in the following order: 2x, 4x, 6x.

Ad Hoc Plus Batch Example

In this example, a job using 10x the queue resources is running. After the job is halfway complete, the same user starts a second job needing 1x the queue resources.

- In the FIFO queue, the 10x job will run until it no longer uses all queue resources (map phase complete, for example), and then the 1x job will start.
- In the Fair queue, the 1x job will start, run, and complete as soon as possible – picking up resources from the 10x job by attrition.

Best Practices for Ordering Policies

- Ordering policies are configured on a per-queue basis, with the default ordering policy set to FIFO. Fairness is usually best for on-demand, interactive, or exploratory workloads, while FIFO can be more efficient for predictable, recurring batch processing. You should segregate these different types of workloads into queues configured with the appropriate ordering policy.
- · In queues supporting both large and small applications, large applications can potentially "starve" (not receive sufficient resources). To avoid this scenario, use different queues for large and small jobs, or use size-based weighting to reduce the natural tendency of the ordering logic to favor smaller applications.
- · Use the Maximum AM Resource Limit scheduler property to restrict the number of concurrent applications running in the queue to avoid a scenario in which too many applications are running simultaneously. Limits on each queue are directly proportional to their queue capacities and user limits. This property is specified as a float, for example: 0.5 = 50%. The default setting is 0.1=10%. This property can be set for all queues by setting the Maximum AM Resource Limit property at the root level, and can also be overridden on a per-queue basis by setting the Maximum AM Resource LimitSetting default Application Master resource limit on page 42property at the queue level.

Configure queue ordering policies

You can configure the property for queue ordering policies (yarn.scheduler.capacity.<queue-path>.ordering-policy) to FIFO or Fair. The default setting is FIFO.

Procedure

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue you want to configure queue ordering policies and select the Edit Queue Properties option.

3. In the Queue Properties dialog-box, select the ordering policy as FIFO or Fair using the Ordering Policy drop-down box .
4. Click Save.

Autoscaling clusters

Autoscaling is a feature that adjusts the capacity of cluster nodes running YARN by automatically increasing or decreasing, or suspending and resuming, the nodes in a host group. You can enable autoscaling based either on a schedule that you define, or the real-time demands of your workloads.

Load-based autoscaling



Attention:

Load-based autoscaling is available for clusters with the following Cloudera Runtime versions:

- 7.2.15 and higher

Load-based auto-scaling is supported in AWS and Azure. In GCP, load-based autoscaling is available as technical preview and is under entitlement. To obtain the required entitlement, contact your Cloudera Account Representative.

Load-based autoscaling suspends and resumes (stops and starts) instances on the cloud provider to increase or decrease capacity for nodes running NodeManagers (for example, the compute host group), based upon YARN's assessment of pending demand and available capacity. Load-based autoscaling can help control costs while providing quick, on-demand cluster capacity when you need it (within a few minutes).

When you configure a load-based autoscaling policy, you choose a minimum and maximum number of nodes for the host group. The maximum number of nodes determines how many instances are provisioned, but these instances are suspended and resumed as the workload demand requires. The policy will not provision instances beyond the maximum range of nodes that you define, regardless of the demand on the cluster. You also define a cooldown period, which is the amount of time in minutes to wait before another autoscaling operation is performed.

Schedule-based autoscaling



Attention:

Schedule-based autoscaling is available for clusters with the following Cloudera Runtime versions:

- 7.2.15 and higher

Schedule-based autoscaling scales the nodes in a host group up or down based upon a schedule that you define. Schedule-based autoscaling is useful if workload demands tend to be high or low on a fairly regular, consistent basis.

When you configure a schedule-based autoscaling policy, you define a target node count, which is the number of nodes that you want to scale up or down to at a particular time. You also select whether or not to repeat the schedule, and finalize the schedule by entering a CRON expression and selecting the desired timezone. When the particular time/date that you define in the CRON expression occurs, the cluster is upscaled or downscaled to your target node count. The time taken to add nodes to a cluster varies by cloud provider and the configuration of the nodes (for example, recipes can have an impact on the time it takes to add a node).



Note: Typically, schedules (especially those configured to scale-up) should be defined in a manner to factor in the time it takes to add new nodes. For example, workloads starting at 9pm will typically set a scale-up schedule for 8:45 pm, assuming it takes around 15 minutes to add a node.

General considerations

Currently, YARN NodeManager is the only service compatible with autoscaling. Because of this, autoscaling is only available by default in the Data Engineering and Data Engineering HA cluster definitions. Autoscaling can

also be configured for clusters with custom templates that include YARN NodeManager and the necessary gateway components.

For autoscaling Cloudera Operational Database clusters, refer to the Autoscaling in public cloud environments and Fast autoscaling in Cloudera Operational Database documentations.

Known Limitations

- Load-based autoscaling does not work with YARN dynamic queues.
- Autoscaling policy creation for Cloudera Data Hub clusters fails during Data Lake rolling upgrade, in case autoscaling was not enabled before starting the Data Lake rolling upgrade.

Autoscaling behavior

Before you define an autoscaling policy, note the following autoscaling behaviors.

General behaviors

- Clusters can perform one upscale or downscale operation at a time.
- A cluster will continue to accept jobs while it is running, regardless of any in-progress upscale or downscale operations.
- Only one autoscale policy type (either load-based or schedule-based) can be configured for a single host group, in a single cluster, at a time.
- Autoscaling is available for host groups with nodes running YARN NodeManager only (and optionally client/GATEWAY components, but not any other service components).
- If there are not enough nodes available to match the requested scale operation, the operation will proceed on however many nodes are available (for example, during a request for a 10 node scale-up, if the cluster loses 1 node, the operation will proceed with scaling-up 9 nodes instead of 10).
- Autoscaling will be disabled if the cluster has any node failures on instances running YARN ResourceManager, or the ClouderaManager node.
- After scaling down, nodes will show up as UNHEALTHY in Cloudera Manager. This is expected. While scaling down, stopped nodes are put into maintenance mode, to suppress alerts.



Attention: Autoscaling is not available for Cloudera Data Hub clusters created with the default Streaming Analytics Light Duty cluster template.



Note: Make sure to create clusters with at least one node in each host group running YARN NodeManagers, for automatic resource configuration to take effect (unless explicitly configuring YARN node-level resources in the cluster template). Once a cluster has been created, the node count for the same can go to 0 (or the minimum can be set to 0 via autoscaling configuration).



Warning: Before performing the FreeIPA upgrade for Cloudera Data Hub clusters where autoscaling is enabled, you must disable autoscaling and start all compute nodes to ensure the healthy state of Cloudera Data Hub clusters. After upgrading FreeIPA, autoscaling can be enabled again. For more information, see the Autoscaling must be stopped before performing FreeIPA upgrade description in the Known issue.



Warning: Autoscaling is not permitted on instances with a DataNode due to the unpredictable behavior of DataNodes during certain operations, such as data shuffling. As these operations can interfere with autoscaling, it is not supported to manually add the DataNode service to a compute node with autoscaling policy.

Cluster management operations with load-based autoscaling enabled

Specific operations are affected as follows:

- Node repair: If the node on which Cloudera Manager is running is selected for repair, then all the stopped nodes need to be selected for repair.

- Cluster stop/start: Cluster stop is supported. During a cluster start, all nodes in the host group will be started and recommissioned.
- Node delete: Node delete (non-forced) is supported, including for STOPPED instances. Forced node delete may require you to End Maintenance (Enable Alerts/Recommission) in Cloudera Manager after performing another upscale operation.
- Retry: Do not attempt retry on an autoscale operation. Instead, see *Manually recovering from load-based scaling failures*. The nature of load-based autoscaling is to automatically try again after some time, based on the load on the cluster. There are a few scenarios where this retry will be adequate to continue cluster operations.

Only a single management operation (including scaling) can be performed on a cluster at a time. For example, if an upgrade is in progress, scaling cannot be performed until the upgrade completes.

EBS versus ephemeral storage



Caution: When you configure a load-based policy, you may receive a warning about using attached EBS volumes over a certain size. While instances are in a STOPPED state (scaled-down), EBS volumes continue to be charged. In order to reduce costs, Cloudera recommends using instances with ephemeral storage (for example, the r5d or m5d instance families) for compute nodes. Ephemeral volumes will also result in better performance of the workloads–they are typically faster, and don't utilize network throughput when writing data.

YARN Decommission Timeout

- Schedule-based downscaling will attempt to remove nodes to reach the target count, regardless of whether the node has running tasks, or data for a running job. You can use YARN DecommissionTimeout to control the leeway available to jobs beyond the scheduled decommission time. For example, with YARN decommission timeout set to one hour, a scheduled downscale will result in the following behavior:
  - Vacant nodes (no running tasks, no tasks previously run for running jobs) will be removed immediately.
  - Other nodes will be marked as decommissioning, and will not accept new work.
  - The remaining nodes will be terminated once any jobs complete, or at the one hour (YARN decommission timeout) mark.

For load-based autoscaling, the YARN DecommissionTimeout should be set to 30 seconds. See Configuring autoscaling for more details.

Autoscaling Role Placement Guidelines

When autoscaling is enabled, ensure that the compute nodes do not run critical services such as HDFS DataNode and ZooKeeper.

Nodes running critical services cannot be stopped by the autoscaling mechanism, which will cause the scale-down operations to fail.

Configuring autoscaling

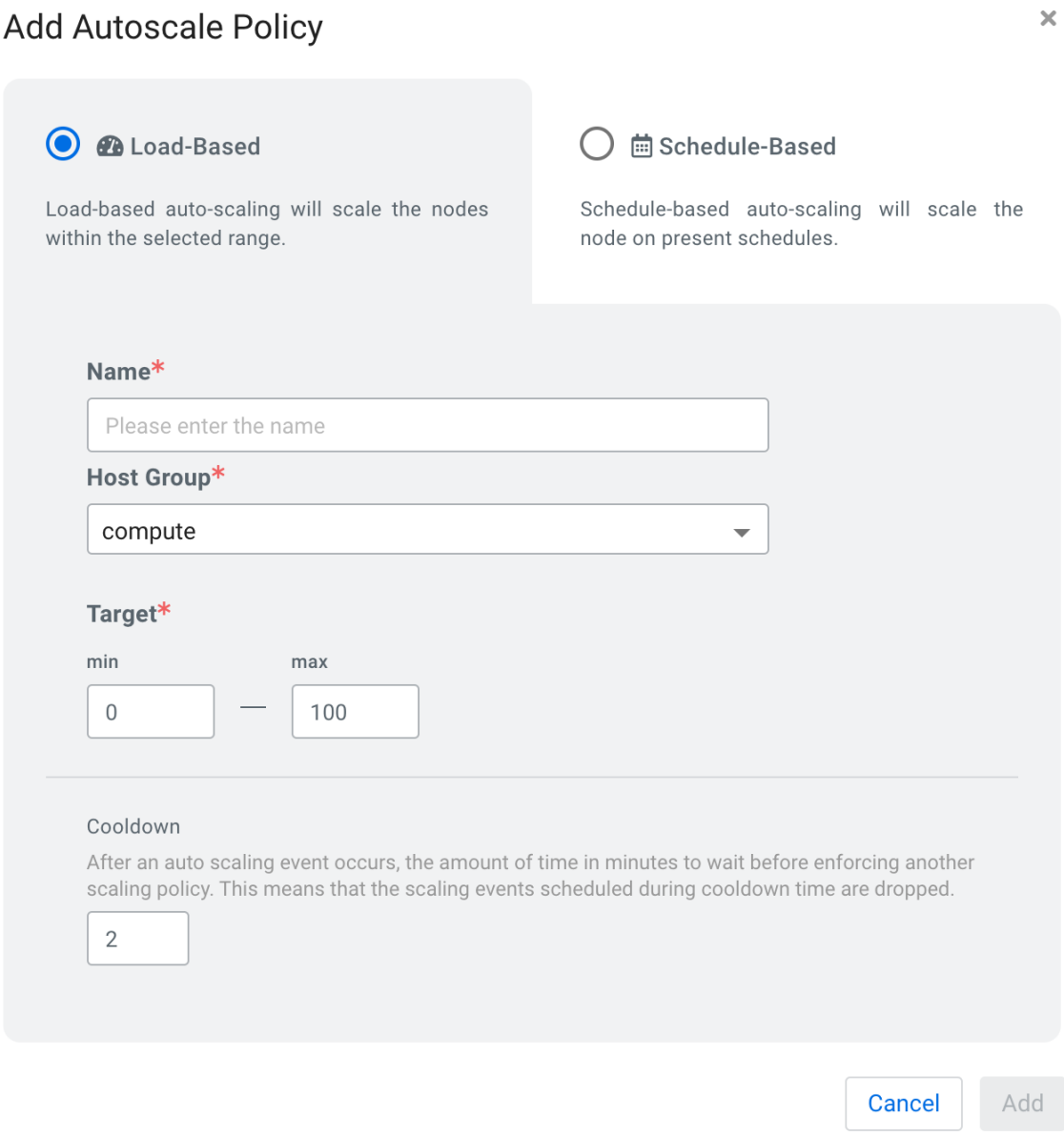
To configure autoscaling, add a load-based or schedule-based policy to a cluster and define the policy parameters.

Before you begin

If you are configuring a load-based autoscaling policy, you must set the YARN Node Decommission Timeout property to 30 seconds. Configure the following property in Cloudera Manager: yarn_resourcemanager_nodemanager _graceful_decommission_timeout_secs

Procedure

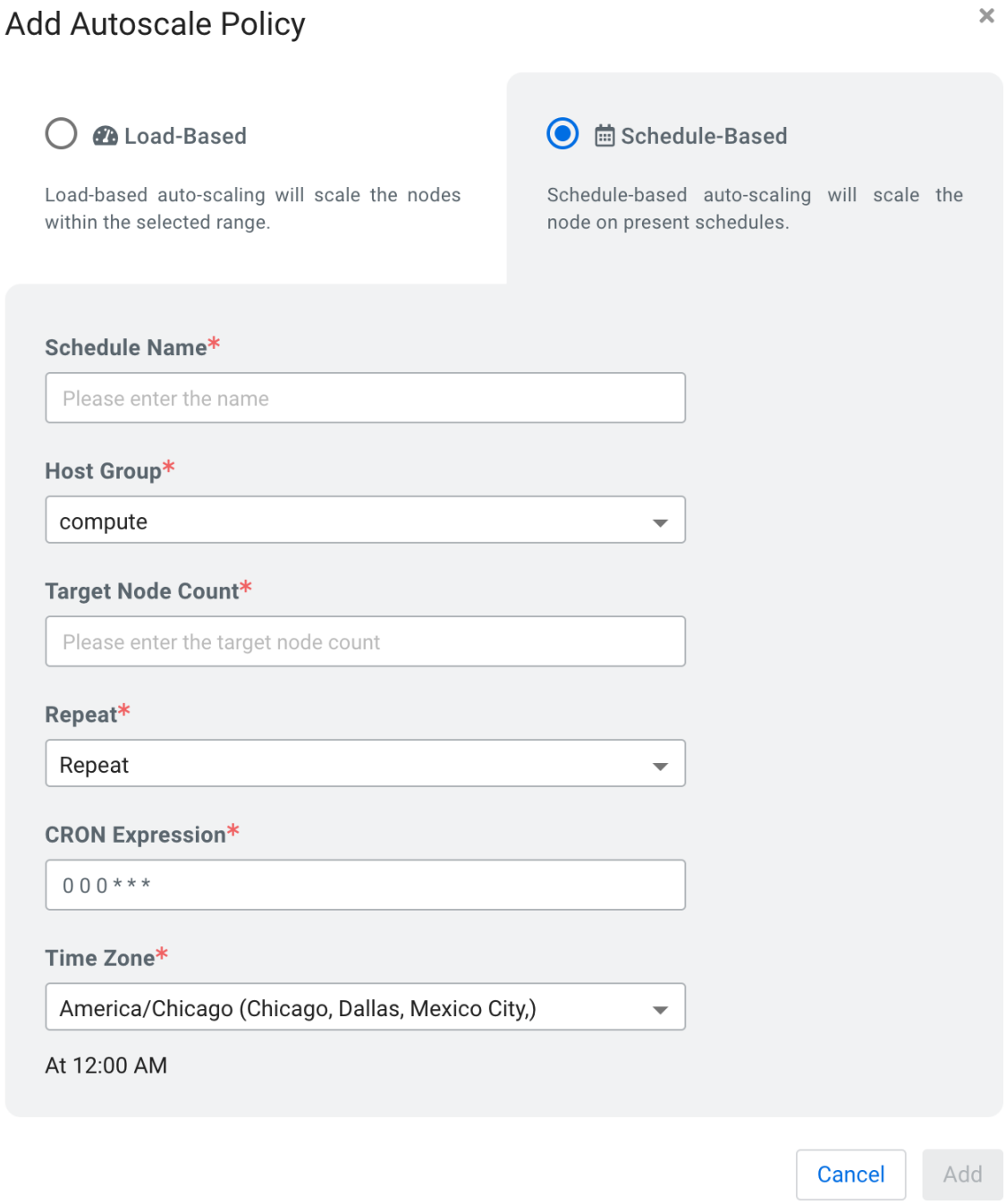
1. From the Cloudera Management Console, click Data Hub Clusters and then select the cluster that you want to add an autoscaling policy to. You can add autoscaling policies to clusters after they have been created, not during the cluster creation process.
2. From the cluster details page, select the Autoscale tab and then click the slider button to enable autoscaling. Autoscaling is disabled by default.
3. Click Add Autoscale Policy and select Load-Based or Schedule-Based.
4. Define the policy parameters.
   - For load-based scaling:



Parameter	Description
Name	Enter a unique name for the policy.

Parameter	Description
Hostgroup	Select the host group that you want to scale. The list of available host groups is determined by which host groups include services that can be scaled.
Target	Enter a minimum and maximum number of nodes for the policy. The maximum number of nodes determines how many instances are provisioned, but these instances are suspended and resumed as the workload demand requires. The policy will not provision instances beyond the maximum range of nodes that you define, regardless of the demand on the cluster.

- For schedule-based scaling:



Parameter	Description
Schedule Name	Enter a unique name for the schedule.
Hostgroup	Select the host group that you want to scale. The list of available host groups is determined by which host groups include services that can be scaled.
Target Node Count	Enter the number of nodes that you want to upscale or downscale the host group to.
Repeat	Currently, only repeating schedules are supported.

Parameter	Description
CRON Expression	Enter a CRON expression string to define the details of the schedule that you want to create.
Time Zone	Select the appropriate timezone on which to base the CRON expression.

5. Click Add.

Dynamic Queue Scheduling

The Dynamic Queue Scheduling feature enables you to dynamically change the system state by reconfiguring the values of properties at a predefined time.

The Dynamic Queue Scheduling feature supports queue-level resource allocation configurations only. Additionally, the feature is only supported in relative and absolute resource allocation mode, and does not include dynamic queues.

The Dynamic Queue Scheduling feature enables you to set a time when a predefined configuration is applied to the YARN Queue Manager system.

Review the following scenario: There are two system states, State-A and State-B, that have to be scheduled for the system. State-A should be used from 8 AM to 8 PM, and State-B should be used from 8 PM to 8 AM. In this case two scheduling rules have to be created:

- Scheduling rule A takes effect at 8 AM and applies Dynamic Configuration State-A to the system.
- Scheduling rule B takes effect at 8 PM and applies Dynamic Configuration State-B to the system.

When a new Dynamic Configuration is created, it is based on the current configuration and enabled by default. Dynamic Configurations can be disabled at any time, either manually or by the system. If a Dynamic Configuration is an invalid configuration, it is disabled automatically by the system and it cannot be enabled manually until it is brought back into a valid state.

Dynamic Configurations do not contain any structural changes such as adding or deleting queues. Moreover, some manual structural changes can invalidate already existing Dynamic Configurations. As a result, existing Dynamic Configurations are validated when any structural changes occur. The following table explains how and when the system validates a Dynamic Configuration and the results of validation.

Table 2: Dynamic Configuration validation

Validation time	Triggering event	What is checked?	Action if Dynamic Configuration is invalid	Action if Dynamic Configuration is valid
Design time	When saving a Dynamic Configuration.	The system checks if the Dynamic Configuration is valid and can be applied.	The Dynamic Configuration is saved in a disabled state and marked as invalid.	Dynamic Configuration is saved in an enabled state.
Run time	When the system begins to apply a Dynamic Configuration.	The system checks if the Dynamic Configuration is valid and can be applied.	Dynamic Configuration is not applied and is disabled.	Dynamic Configuration is applied to the system.

In addition to the listed automatic validation, manual validation is required after queue configuration changes. This validation is not triggered automatically, but the UI will prompt you to perform Dynamic Configuration revalidation when you make any queue configuration changes. For more information, see *Revalidate Dynamic Configurations.*

Creating a new Dynamic Configuration

In Cloudera Manager, using the YARN Queue Manager UI you can create new Dynamic Configurations that will be used by the Dynamic Queue Scheduling feature.

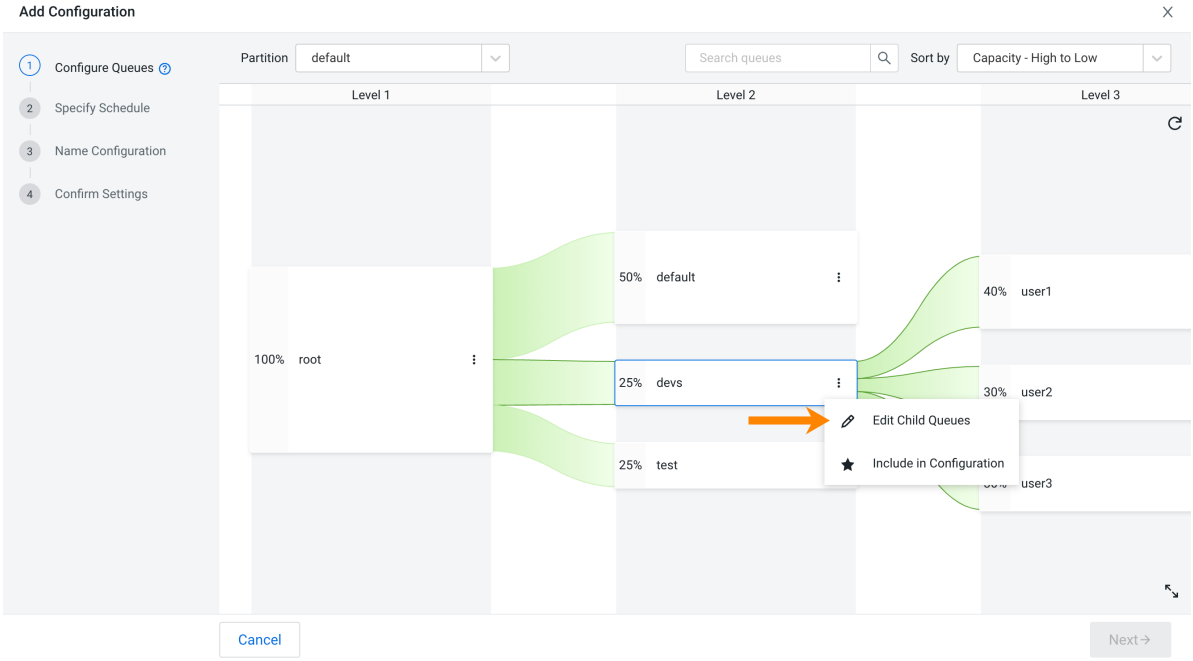
Procedure

1. In Cloudera Manager, navigate to Clusters YARN Queue Manager UI .
2. Click Schedule.
3. Click +Add.
   The **Add Configuration** window is displayed with the current configuration setting.
4. Configure Queues.

   When configuring queues for the Dynamic Configuration, you can either change a queue's configuration or include its current values.

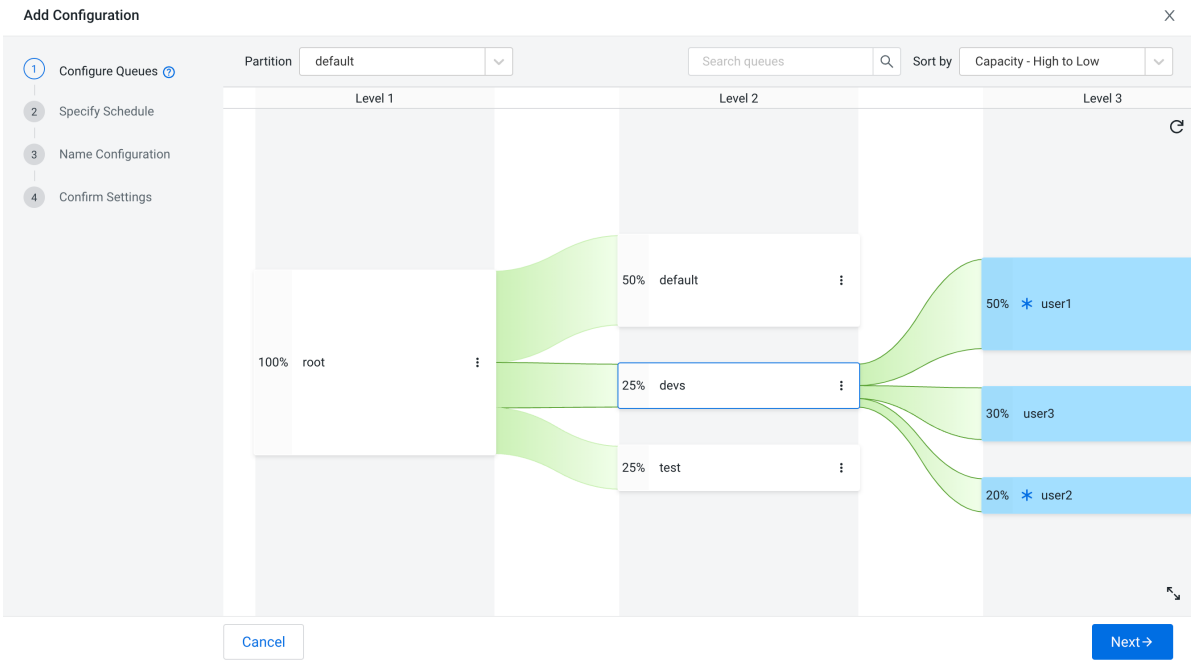
   a) Changing queue configuration:

   Edit child queues by clicking the three vertical dots next to the parent queue's name and select Edit Child Queue.



After you edit the child queues, they are included in the Dynamic Configuration which is indicated by a blue background. In relative resource allocation mode if one queue is included in the configuration at a certain level, all other queues on that level (siblings) are also included.

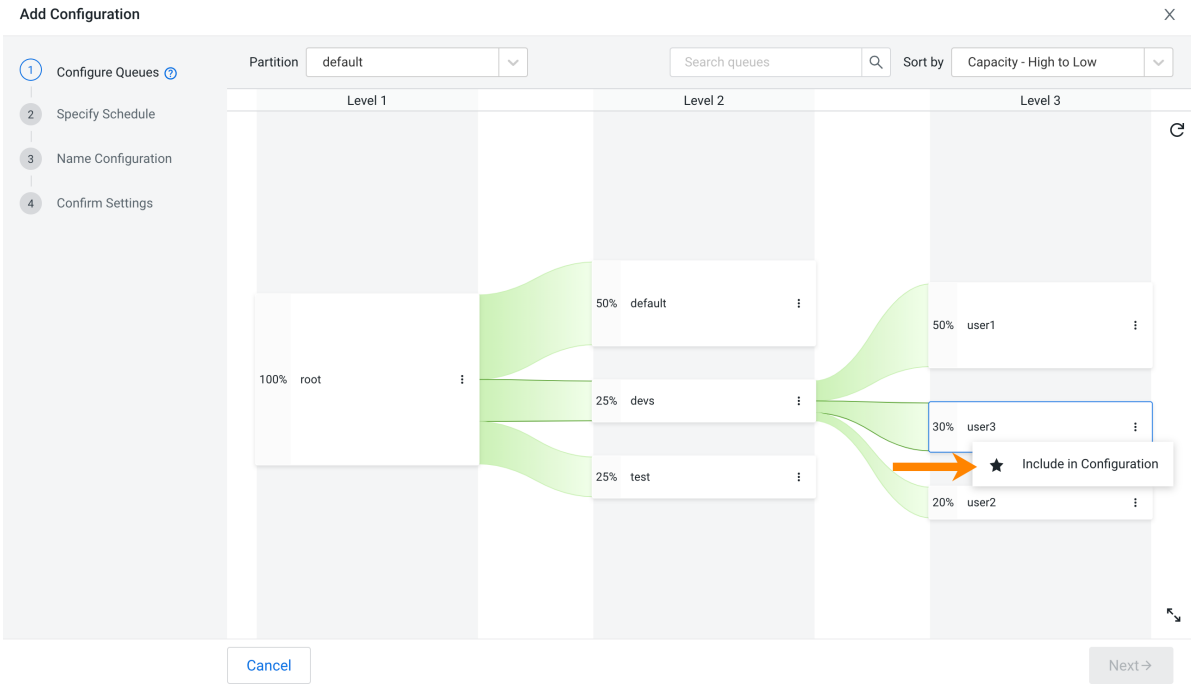
A blue asterisk next to the queue’s name indicates that you have changed the properties of that queue to create the Dynamic Configuration.



In this example the user1 and user2 queue's capacity properties were changed from 40% to 50% and from 30% to 20% respectively. The user3 queue has the same setting as it had in the configuration that is the base of the Dynamic Configuration that is currently being created.

b) Including the current values in the dynamic configuration:

Click the three vertical dots next to the queue which you want to include in the configuration and click Include in Configuration.



As a result, in relative resource allocation mode the selected queue and its sibling queues are included in the configuration with their current resource configuration. In absolute resource allocation mode, only the selected queue is added to the configuration.

As there was no change in the queue's resource configuration, no asterisk was added next to the queue's name.

5. Revert Changes.

   If you want to revert changes you made to a queue, click the three vertical dots next to the name of the queue and select Revert Changes.

   As a result, in relative resource allocation mode all changes on the selected queue and its sibling queues are reverted. In absolute resource allocation mode, only the changes of the selected queue are reverted.

6. Remove from Configuration.

   If you want to remove a queue with its current resource configuration from the Dynamic Configuration, click the three vertical dots next to the queue's name and click Remove from Configuration.

   As a result, in relative resource allocation mode the selected queue and its sibling queues are removed from the configuration. In absolute resource allocation mode, only the selected queue is removed to the configuration

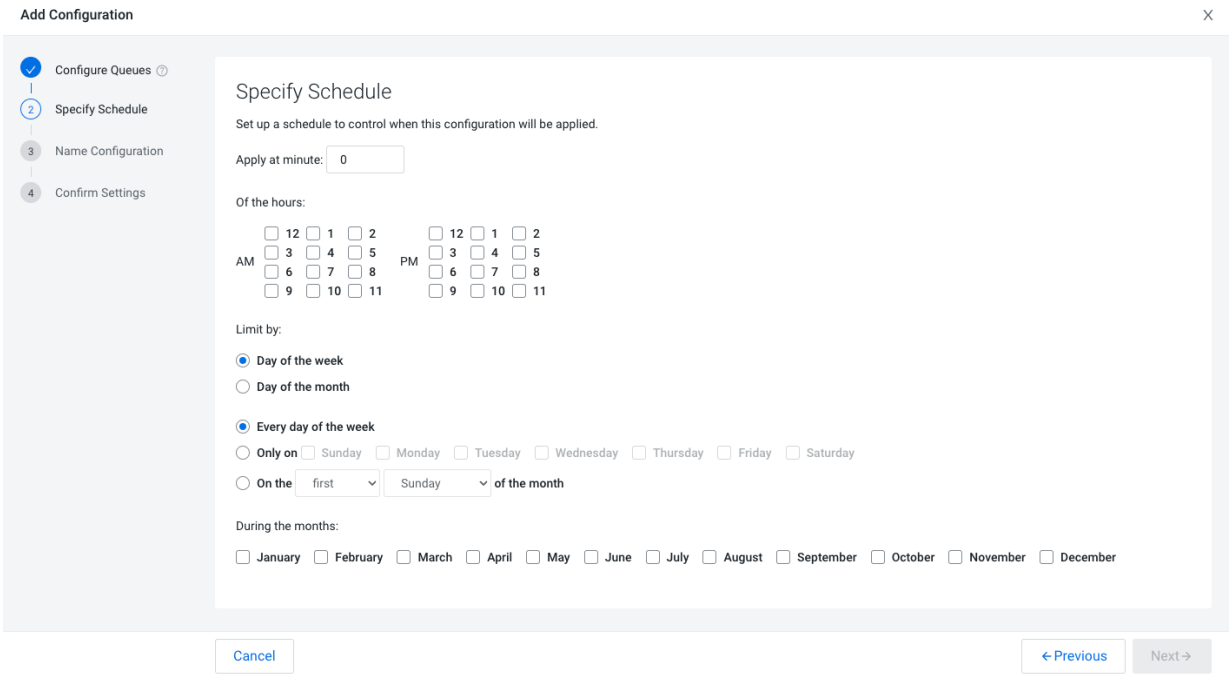
7. Click Next.

8. Specify Schedule.

   Important: When specifying schedule the date and time that is taken into account is the server date and time, not the local date and time of the user.

   Set up a schedule to control how frequently the Dynamic Configuration is applied.

   The time and frequency can be configured using the various options available on the page. Once an option is selected a note appears on the top of the window that tells you the exact time and frequency when the schedule is applied.



   Using the Apply at minute: and Of the hours: options, you can set the exact time when a configuration should be applied. The Of the hours: section allows multiple values to be selected simultaneously. The Apply at minute: option however only accepts a single value. This means that while you can specify a schedule to apply at 6:00 AM and at 8:00 AM, you can't specify a schedule to apply for example at 6:00 AM and 8:15 AM. If you use the default 0 value for the Apply at minute: option, and you do not select any hours, then the configuration will be applied at every hour.

   You can configure the schedule on a weekly or a monthly basis, by selecting either Limit by: Day of the week or Limit by: Day of the month.

   If you select Day of the week, you can choose from the following options:

- Every day of the week
- Only on: The configuration is applied on selected day or days of the week.
- On the *[***FIRST-LAST***] [***MON-SUN****]* of the month: The configuration is applied on the selected day.

If you select Day of the month, you can choose from the following options:

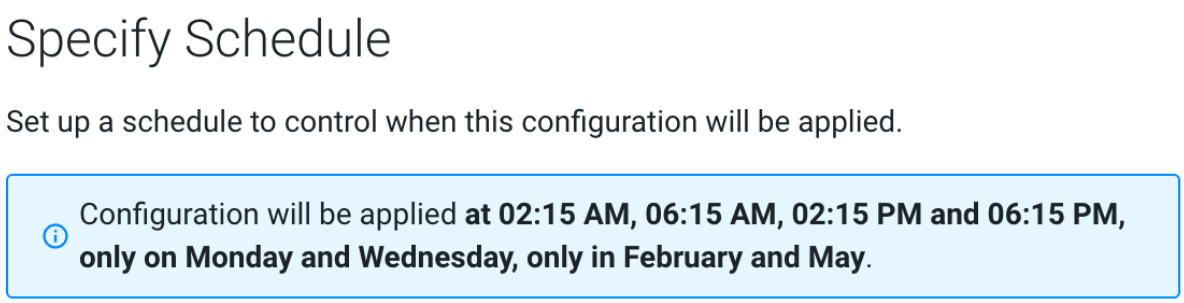
- Every day of the month
- Every # days starting on day # of the month: The configuration is applied on the specified days based on the configured starting day and frequency.
- On day(s): The configuration is applied on the specified days. This option accepts integers 1-31. You can specify a single integer to select a single day, specify a range to select multiple days, or you can also add a combination of the two. For example, 1-5, 8, 11-13.
- # day before the last day of the month
- On the weekday nearest day # of the month
- On the last weekday of the month

In addition, you can specify month or months when the schedule is applied. If you do not specify any month, the schedule is applied in every month.

Consider the following example:

- Apply at minute: 15
- Of the hours: AM 2, 6 and PM 2, 6
- Limit by: Day of the week
- Only on: Monday, Wednesday
- During the months: February, March

Then the configuration will be applied at 02:15 AM, 06:15 AM, 02:15 PM and 06:15 PM, only on Monday and Wednesday, only in February and May. The UI will display the schedule:



9. Click Next.
10. Name Configuration.

    Provide a unique name for the Dynamic Configuration and click Next.
11. Confirm Settings.

    Review and confirm the scheduling rule and click Finish.

Managing Dynamic Configurations

In Cloudera Manager using the Queue Manager UI you can manage the Dynamic Configuration you have created for the Dynamic Queue Scheduling feature.

Procedure

1. In Cloudera Manager, click Clusters YARN Queue Manager UI .

2. Select Schedule.
3. Find the Dynamic Configuration you want to manage from the list.
4. Click the three vertical dots for the scheduling rule.
5. Select an action:
   - View: Displays the configuration details of the Dynamic Configuration.You can review the name, schedule, and queue configuration of the Dynamic Configuration.
   - Disable/Enable: Allows you to enable or disable the Dynamic Configuration. If you disable a Dynamic Configuration it will not be applied to your system at its scheduled time. Later, you can enable it again.

     Note: Invalid Dynamic Configurations are disabled automatically and cannot be enabled manually until they are brought back into a valid state.
   - View conflict: Available if the Dynamic Configuration is conflicted with any previously created Dynamic Configuration. Use this action to review the conflicts or to accept the Dynamic Configuration with conflicts.
   - Edit: Allows you to edit details of the Dynamic Configuration, such as queue configuration, schedule, or name.
   - Delete: Allows you to delete the Dynamic Configuration.
   - Activate Now: Allows you to activate the queue configuration of this Dynamic Configuration immediately.

How to read the Configurations table

In the Queue Manager UI you can view all of your Dynamic Configurations on one page. Learning about this page can help you manage the Dynamic Queue Scheduling feature.

The schedule overview page is displayed in Cloudera Manager if you select the Queue Manager UI and then go to the Schedule tab. It displays a table with the following columns:

Table 3: Schedule table

Column name	Description
Status	The status of the Dynamic Configuration: •  Valid and enabled •  Valid and disabled •  Invalid •  Disabled due to scheduling conflict •  Enabled with scheduling conflict •  The resource allocation mode of the Dynamic Configuration differs from the currently used resource allocation mode. As a result the Dynamic Configuration will fail when executed.
Name	The name of the Dynamic Configuration.
Scheduled At	Information about the time at which the Dynamic Configuration is scheduled to be applied to the system.

Handling Dynamic Configuration conflicts

A dynamic queue configuration related scheduling conflict occurs if two or more Dynamic Configurations are scheduled to be applied at the same time. You can either enable Dynamic Configuration with conflicts or modify the conflicting Dynamic Configuration to resolve conflicts.

About this task

When a Dynamic Configuration is created with a schedule that conflicts with one or more other Dynamic Configuration, the new Dynamic Configuration will be saved in a disabled with conflict state.



Note: Although recurrence patterns of Dynamic Configurations can be open ended, scheduling conflicts are detected only one year ahead of time.

Procedure

1. Find the Dynamic Configuration that is conflicted.
2. Click the three vertical dots in the line of the conflicted Dynamic Configuration.
3. Select View Conflicts.

   All conflicting Dynamic Configurations and conflicting times are displayed.
4. Review the conflicts and perform one of the following:
   - Accept the existing conflicts:
     a. Select Accept all conflicts.
     b. Click Enable.

        The system will execute the conflicted Dynamic Configurations one after another without any predetermined order.
   - Resolve conflicts by modifying the conflicting schedule:
     a. Click Cancel.
     b. Click the three vertical dots in the line of the conflicted Dynamic Configuration.
     c. Select Edit.
     d. Modify the Dynamic Configuration to resolve the conflict.
     e. Click Save.

Revalidating Dynamic Configurations

Dynamic Configurations need to be revalidated after queue configuration changes. You can revalidate Dynamic Configurations using the YARN Queue Manager UI.

About this task

When changing the queue configuration, on the bottom of the queue's configuration dialog box the following message is displayed to remind you that Dynamic Configuration revalidation needs to be performed:

```
Changes you make here may invalidate dynamic configurations, so you will nee
d to revalidate them. This can be performed on the Schedule tab.
```

In addition a red dot appears next to the Scheduler tab, indication that some action needs to be taken related to Dynamic Queue Scheduling.



Note: The revalidation of Dynamic Configurations can take a significant amount of time if you have a higher number of queues. Until the revalidation is done you cannot use the Schedule tab of the YARN Queue Manager UI.

Procedure

1. In Cloudera Manager nevigate to the YARN Queue Manager UI.
2. Click Schedule.

   After any queue change the following message is displayed on the Schedule page:

   ```
   Revalidation Required: All dynamic configuration must be revalidated due
    to queue changes. Please click Revalidate to proceed.
   ```

3. Click Revalidate.

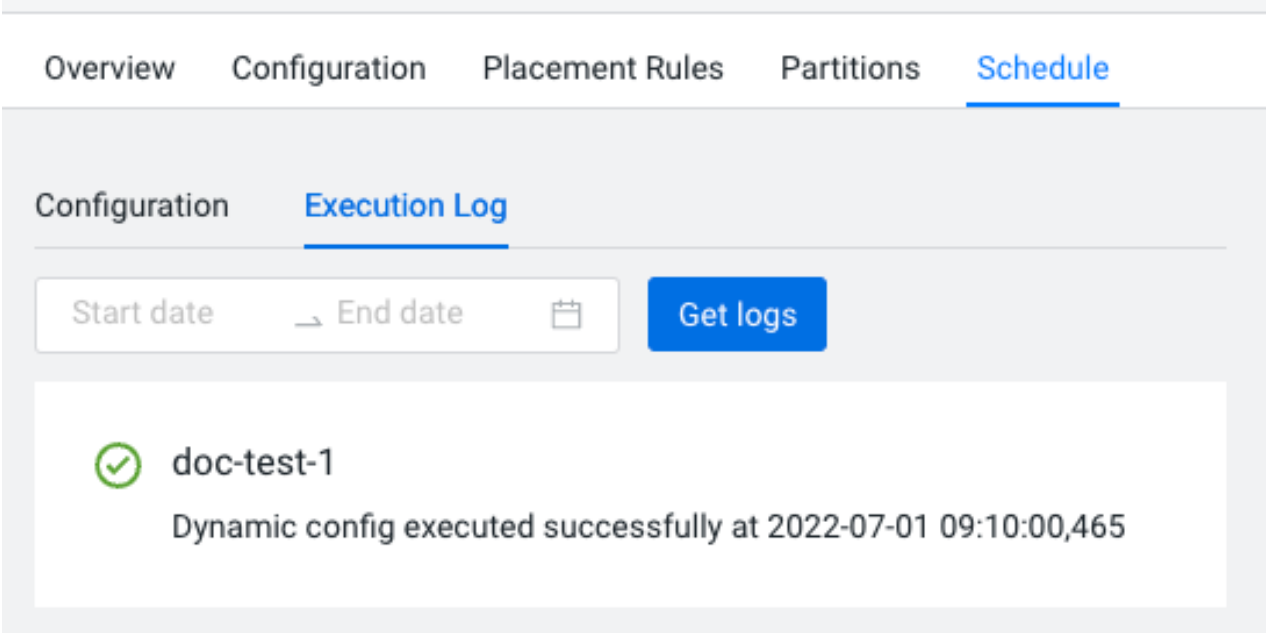
   As a result an orange dot next to the Schedule tab, and Revalidation in progress status text appear.
4. Wait until revalidation is finished.

   You do not have to stay on the Schedule page until the Revalidation is done. Once the orange dot next to the Schedule tab disappears , the revalidation is finished.
5. Check if any of the Dynamic Configuration became disabled.
6. Perform one of the following actions for the disabled Dynamic Configurations:
   - Update the queue configuration to make the Dynamic Configuration valid
   - Update the Dynamic Configuration to make it valid
   - Delete the invalid Dynamic Configuration

Dynamic Configurations execution log

The execution log page displays all Dynamic Configurations that are executed, either according to their schedule or manually. The log provides information if those Dynamic Configurations were applied successfully or failed.

To view the execution log navigate to YARN Queue Manager UI Schedule Execution Log:



The execution log will display all executed Dynamic Configuration providing the following information:

- Name of the Dynamic Configuration
- If the Dynamic Configuration was applied successfully or failed
- The exact date and time of the execution

Note: The Dynamic Configuration name is not displayed correctly when it is manually activated.

You can also set a Start date and End date to view the logs of a specific time period. When they are not specified the default time period is 7 days, meaning that the execution logs for the last 7 days are displayed.

Managing placement rules

Placement rules can define the logic that is taken into account when specifying which queue should be used for a submitted job. These predefined rules enable you to submit jobs without specifying the queue name at the time of job submission.

There are two kinds of queue to which jobs can be submitted:

- Static queues: Queues that always exist and were defined by the user using the Queue Manager UI (or configuration files).
- Dynamic queues: Queues that are created dynamically when jobs are submitted to them. If the YARN service is restarted they are automatically deleted. To learn more about dynamic queues, see *Manage dynamic queues*.

Placement rules enable you to define the logic that applies when a job is submitted to specify which queue should be used for the submitted job. That enables you to submit a job without defining a target queue or to even override the target queue that was specified by the submitter during the job submission.

By default, placement rules are taken into account only if no target queue is specified during a job submission or if the specified target queue is provided as “default”. To change this behavior, see *Enable override of default queue mappings*.

Placement rules are evaluated in the order in which they appear in the placement rule list. When a job is submitted and placement rules have to be taken into account, the rules are evaluated, and the first matching rule is used to determine the queue in which the job runs.

If there is no placement rule and no target queue was specified during the job submission, then the job is submitted to the default queue of the scheduler.

If the target queue of a placement rule does not exist or it cannot be created, the configured fallback action is performed. You can configure the fallback action for each placement rule, and it can have the following values:

- Skip: Ignore the current rule and proceed to the next.
- PlaceDefault: Place the application to the default queue root.default (unless it is overridden to something else).
- Reject: Reject the submission.

If no target queue was specified during the job submission and no placement rule matches the job, then the job is submitted to the default queue of the scheduler.

By default, if an invalid queue was specified during a job submission, the submission is rejected. To change this behavior, see *Enable override of default queue mappings*.

Related Information

Managing dynamic queues

Placement rule policies

When creating a placement rule you have to set its policy to define the logic that is taken into account when specifying which queue should be used for a submitted job. There are many predefined policies you can choose from, or you can create a custom policy.



Important: When referring to a queue, Cloudera recommends to always provide the parent queue as well. Although, in Capacity Scheduler you can refer to a queue only by its leaf queue name, it can lead to issues if there are more leaf queues with the same name. Providing the parent queue ensures that the reference is translated to a fully qualified path, that is, there will be no ambiguity.

The following table lists the names of the policies, the options displayed in the Placement Rule creation dialog box in the Queue Manager UI, and their detailed description:

Table 4: Placement rule policies

Policy	Queue Manager UI	Description
user	Place the application into a queue named for the user.	Places the application into a queue which matches the username of the submitter.
primaryGroup	Place the application into a queue named for the user's primary group.	Places the application into a queue which matches the primary group of the submitter.
primaryGroupUser	Place the application into a queue named for the user that is the child of a queue named for the user's primary group.	Places the application into the queue hierarchy [parentQueue].<primaryGroup>.<userName>. The parentQueue is optional.
secondaryGroup	Place the application into a queue named for the user's secondary group.	Places the application into a queue which matches the secondary group of the submitter.
secondaryGroupUser	Place the application into a queue named for the user that is the child of a queue named for the user's secondary group.	Places the application into the queue hierarchy [parentQueue].<secondaryGroup>.<user Name>. The parentQueue is optional.
applicationName	Place the application into a queue named for the application.	Places the application into a queue which matches the name of the application. **Important:** It is case-sensitive, white spaces are not removed.
specified	Place the application in the queue specified at the run time.	Places the application to the queue that was defined during submission.
reject	Reject the application.	Rejects the submission.
defaultQueue	Place the application into the default queue.	Places the application into the default queue root.default or to its overwritten value.
setDefaultQueue	Set the default queue to:	Changes the default queue from root.default. This policy does not permanently change the default queue. It is always root.default when the evaluation begins when an application is submitted. However, the adjusted default queue remains in effect until the placement rules evaluation is completed. **Important:** • This policy rule does not place the application in the specified queue. • Fallback will not be triggered, no matter what the selection is.
custom	Use the following custom policy:	Enables the user to use custom placement strings.

Using the custom placement policy

The custom placement policy can describe other policies with the appropriate variable placeholders. For example, primaryGroupUser with the parent queue root.groups can be expressed as: root.groups.%primary_group.%user

The *Custom policy variables* table described what variables are available if you intend to use the custom policy.

Internally, the tool populates certain variables with appropriate values. These can be used if custom mapping policy is selected. The placement rules evaluation engine does only minimal verification when it comes to replacing them. Therefore it is your responsibility to provide the correct string.

Table 5: Custom policy variables

Variable	Meaning
%application	The name of the submitted application.
%default	The default queue of the scheduler.
%primary_group	Primary group of the submitter.
%secondary_group	Secondary (supplementary) group of the submitter.
%specified	Contains the queue that the submitter defined. It is a standalone variable, do not combine it with other custom variables or paths. If the specified target queue is default this variable will not be set. If the target queue is the default queue, it should be specified with the root.default parent path.
%user	The user who submitted the application.

If you submit a MapReduce application to a queue root.users.mrjobs, the value of %specified will be set to root.users.mrjobs.

Few predefined policies can be achieved with the custom policy. For example, instead of using the specified policy, you can use the custom policy and set the customPlacement field to %specified.

However, you have more control over the policy if you use custom instead of the predefined one. That is because you can also append or prepend an extra string to these variables. As a result, for example, the following setting is possible:root.groups.%primary_group.admins.%user.longrunningjobs

The string must be resolved to a valid queue path to have a proper placement.

Differences between legacy modes and weight mode

There are some cases when the legacy resource allocation modes (absolute and relative mode) behave differently than weight mode.

The create flag

- Legacy modes: It has no effect if the parent is not managed.
- Weight mode: It applies to all parent queues. However, if yarn.scheduler.capacity.<queue-path>.auto-queue-creation-v2.enabled is set to false, then the queue will not be created.

The nested rules primaryGroupUser and secondaryGroupUser

- Legacy modes: They expect the parent queues to exist, meaning they cannot be created automatically. More specifically, when you use primaryGroupUser it will result in a queue path similar to root.<primaryGroup>.<userName> and the root.<primaryGroup> parent queue must exist. It can be a managed parent in order to have the userName leaf created automatically, but the parent still has to be created manually.
- Weight mode: As long as the parent allows dynamic queues to be created, there is no limitations. The requested queues will be created.

How to read the Placement Rules table

In the Queue Manager UI you can view all of your placement rules on one page. Learning about this page can help you manage your placement rules according to your needs.

The placement rules overview page is displayed in Cloudera Manager if you select the Queue Manager UI and then go to the Placement Rules tab. It displays a table with the following columns:

Table 6: Placement rules overview page

Column Name	Description
Order	Placement rules are evaluated from top to bottom. This column provides the order of the placement rules.
Type	Indicates to the engine what the current rule should be matched against: application, user, or group. Values: Application, Users, Group
Match	The string to match, or an asterisk "*" which means "all". For example, if the type is User and this string is "hadoop" then the rule will only be evaluated if the submitter user is "hadoop". The "*" does not work with groups.
Policy	Pre-defined or custom policies which defines where the application should be placed. Values: User, PrimaryGroup, PrimaryGroupUser, SecondaryGroup, SecondaryGroupUser, ApplicationName, Specified, Reject, DefaultQueue, SetDefaultQueue, Custom
Parent Queue	In the case of User, PrimaryGroup, PrimaryGroupUser, SecondaryGroup, and SecondaryGroupUser policies, this tells the engine where the matching queue should be looked for. For example, if the policy is PrimaryGroup, parent is root.groups and the submitted's group is "admin", then the resulting queue will be: root.groups.admin
Custom Value	It can have two types of values: • In the case of the SetDefaultQueue policy: The default queue will change from root.default to the value of this field. • In the case of the Custom policy: The value of this field will be evaluated directly by the placement rules evaluation engine, which means that various placeholders such as %application or %primary_group will be replaced with their respective values.
Create Queue?	It sets the create flag and it works differently in weight and in legacy mode. If it is set to No, the target queue determined by the placement policy will not be created if it does not exist. That means that auto dynamic child creation will not happen. However, even if it is set to Yes it is still not guaranteed that the queue will be created. You also have to ensure that for the specified parent queue the auto dynamic child creation feature is enabled. The following policies do not support the create flag: • reject • setDefaultQueue • defaultQueue
Fallback	If the target queue does not exist or it cannot be created, it defines a fallback action. Values: Skip, PlaceDefault, Reject

Column Name	Description
Actions	Click on the applicable icon to perform actions on the placement rules: • View Rule • Edit Rule • Delete Rule

Creating placement rules

You can create placement rules using the YARN Queue Manager UI in Cloudera Manager.

Before you begin

Placement rule that uses static queue

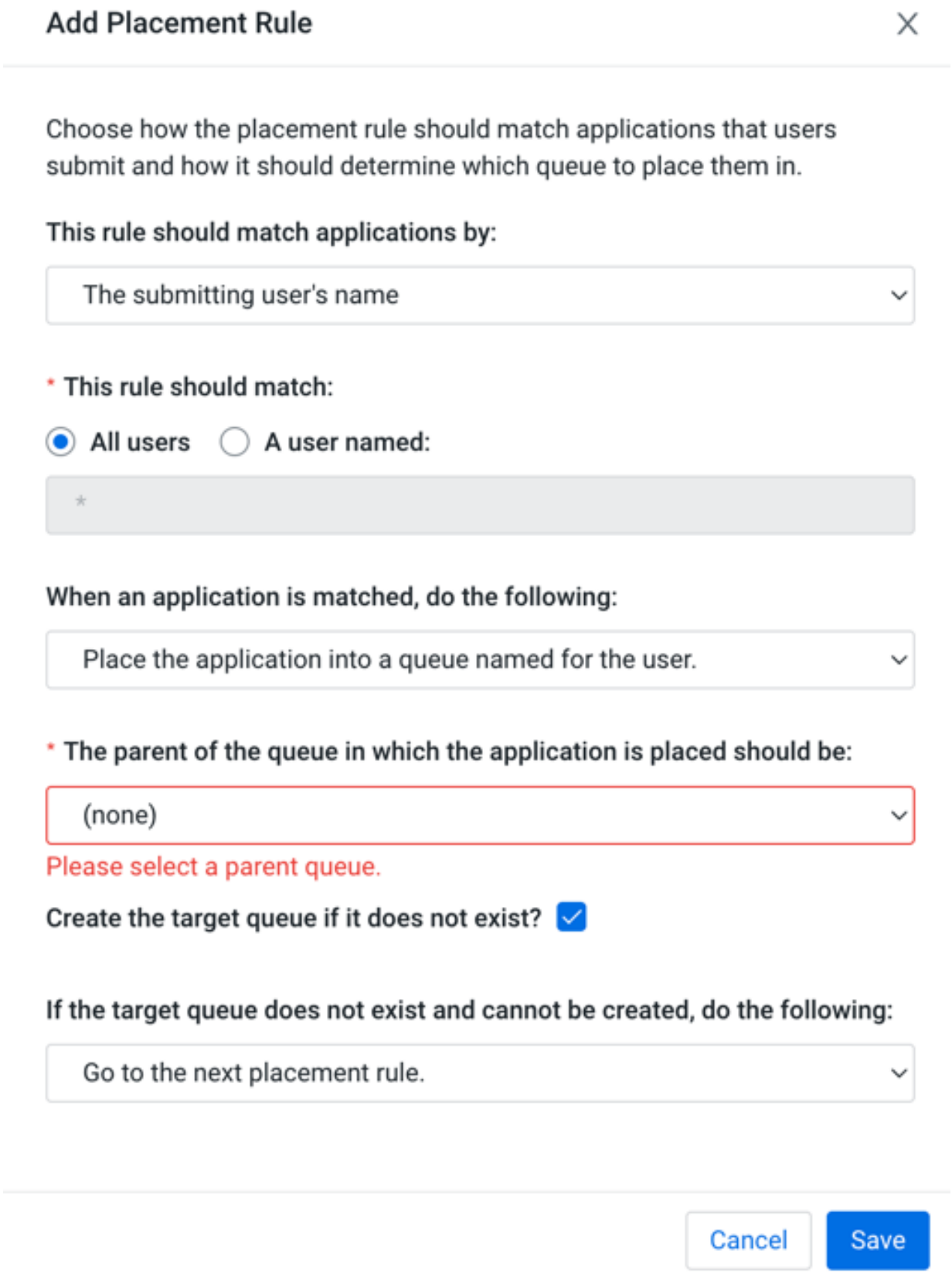
Create the target leaf queue before creating the placement rule that uses it. When creating the placement rule the UI will display all the existing leaf queues.

Placement rule that uses dynamically created queues

Enable the dynamic child creation feature for the target parent queue before creating the placement rule that uses it. When creating the rule the UI will display all the existing queues as target parent queue options but if the dynamic child creation feature is not enabled for the selected queue, a warning message will be displayed and you cannot create the placement rule. For more information, see *Manage dynamic queues*.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Navigate to Placement Rules.
3. Click +Add.
   The Add Placement Rules dialog box is displayed with the default settings or with settings that are the exact replica of the last placement rule that you have created.



4. Use the The rule should match application by property to set the application parameter that should match with the rule.

5. Use the This rule should match field to set the value of the matching application parameter that should match with the rule.
6. Set what the rule should do when an application is matched.

   When an application is matched, do the following: Sets the placement rule policy. The following options are available:

   - Place the application into a queue named for the user.
   - Place the application into a queue named for the user's primary group.
   - Place the application into a queue named for the user that is the child of a queue named for the user's primary group.
   - Place the application into a queue named for the user's secondary group.
   - Place the application into a queue named for the user that is the child of a queue named for the user's secondary group.
   - Place the application into a queue named for the application.
   - Place the application in the queue specified at run time.
   - Place the application into the default queue.
   - Reject the application.
   - Set the default queue to:
   - Use the following custom policy:
7. Set additional configuration for the placement rule.

   Depending on the selected policy, different options become available. These are as follows:

Table 7: Policy specification

Selected policy	Additional configuration
• Place the application into a queue named for the user. • Place the application into a queue named for the user's primary group. • Place the application into a queue named for the user that is the child of a queue named for the user's primary group. • Place the application into a queue named for the user's secondary group. • Place the application into a queue named for the user that is the child of a queue named for the user's secondary group. • Place the application into a queue named for the application.	The The parent of the queue in which the application is placed should be dropdown list appears. Select the parent of the queue to which the job should be submitted.  **Important:** Cloudera recommends to always set the parent queue when it is an available property, even if it is only optional. In this way issues caused by leaf queues with the same name can be avoided.
• Place the application in the queue specified at run time. • Place the application into the default queue. • Reject the application.	No additional configuration required.
Set the default queue to:	A dropdown list appears. Select which available queue you want to make the default queue.
Use the following custom policy:	A text box appears in which you can provide the custom policy you want to use.

8. If you want the target queue to be created, if it does not exist at the time of the job submission, select the Create the target queue if it does not exist? checkbox. To enable this feature you have to set a parent queue in Step 7.

    **Note:** You have to enable the dynamic child creation feature for the parent queue you select if you want the target queue to be created if it does not exist at the time of the job submission.
9. Use the If the target queue does not exist and cannot be created, do the following property to set the fallback action.

   The following fallback actions are available:

   - Go to the next placement rule.
   - Place the application into the default queue.

- Reject the application.

10. Check you placement rule settings.
11. Click OK.
12. Provide a description of the change and then click OK.

Results

The rule is added to the bottom of the placement rule list and becomes the last rule to be evaluated.

Related Information

Managing dynamic queues

Example - Placement rules creation

You have to set up your placement rules to fulfill your specific need. In this example a cluster is shared among developers, QA engineers and test developers and there are nine placement logics that you want to set up.

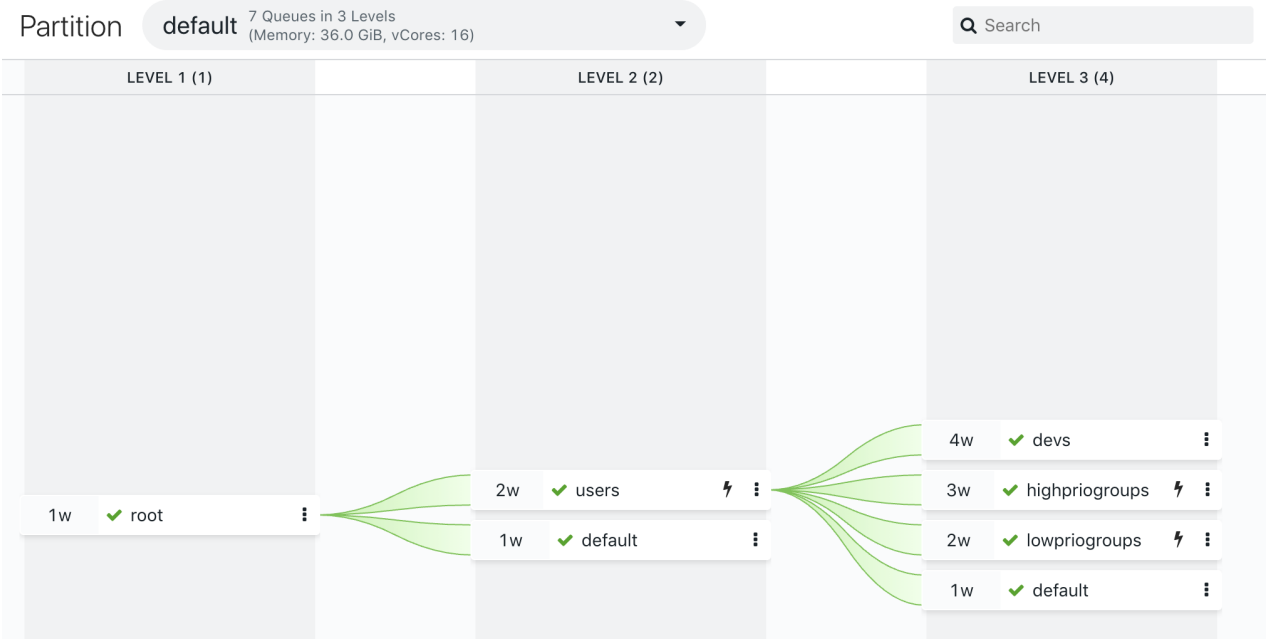
The following is the placement logic that has to be achieved when creating placement rules:

1. If the user belongs to the devs primary group, the application should be placed to root.users.devs. This is reserved for developers.
2. If the user belongs to the qa primary group, then the application should go to root.users.lowpriogroups.<primaryGroup>. These queues have lower capacities and are intended for testers.
3. If the user belongs to the qa-dev primary group, then the application should go to root.users.highpriogroups.<primaryGroup>. These queues have higher capacities and are intended for test developers.
4. Place the application into the queue which matches the user name.
5. If there is no such queue, take the queue from the application submission context, but the queue should not be created if it does not exist and the parent is managed.
6. If none of the above matches, then the application should be placed into the root.default queue.
7. If the default placement fails , change the default queue to root.users.default.
8. Try a placement to the default queue again.
9. If that fails, reject the submission altogether.

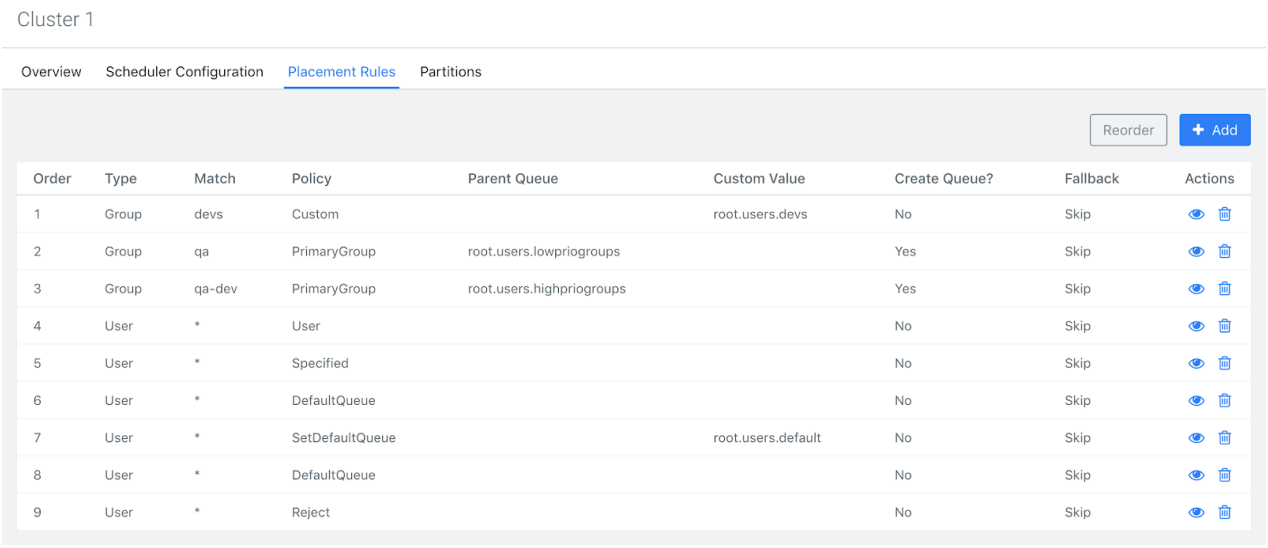
Using the Queue Manager UI, this logic can be achieved in the following way:

Queue hierarchy

Queue with bolt signs next to their names are auto dynamic child creation enabled parents.



Placement rules overview



Reordering placement rules

Placement rules are evaluated in the order in which they appear in the placement rule list. When a job is submitted, the rules are evaluated, and the first matching rule is used to determine the queue in which the job is run.

About this task

When a job is submitted, the rules are evaluated from top to bottom, and the first matching rule is used to determine the queue in which the job is run.

If a rule is always satisfied, subsequent rules are not evaluated. By default, placement rules are ordered in the order they were added; the rule added first appears first. You can easily reorder rules.

Procedure

1. In Cloudera Manager, select YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Go to the Placement Rules tab.
   A list of placement rules is displayed.
3. Click Reorder.

   The Reorder option is only available if you have at least two placement rules.
4. Click Move Up and Move Down arrow buttons in a rule row.
5. Click Save Reorder.

Editing placement rules

The Queue Manager UI enables you to edit previously created placement rules.

Procedure

1. In Cloudera Manager, select YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Go to the Placement Rules tab.
   A list of placement rules is displayed.

3. In the Action column click on the Edit rule icon in the row of the placement rule you want to edit.
   The Edit Placement Rule dialog box is displayed.
4. Edit the placement rule.
5. Click Save.

Deleting placement rules

The YARN Queue Manager UI enables you to delete previously created placement rules. If you want to delete queues associated with placement rules, you first have to delete its associated placement rules.

Procedure

1. In Cloudera Manager, select YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Placement Rules tab.
   A list of placement rules is displayed.
3. In the Action column click on the Bin icon in the row of the placement rule you want to delete.
4. Click Save.

Enabling override of default queue mappings

By default, placement rules are considered only if a target queue is not specified during a job submission.You can change that behaviour to take placement rules into account whether a target queue is specified at job submission or not.

About this task

The yarn.scheduler.capacity.queue-mappings-override.enable property controls when placement rules are considered. In the YARN Queue Manager UI, this property is called Override Queue Mapping. By default, the property is set to false, which means the feature is disabled and placement rules cannot override the target queue that was specified at job submission.

The following table shows how it is specified which queue should be used for a job in different scenarios:

Table 8: Target queue speficiation scenarios

Override Queue Mapping	Target queue is specified at job submission?	Placement rules exist?	End result
Disabled (set to false)	Yes	Yes	Job is submitted to the queue that was specified by the submitter.
Disabled (set to false)	Yes	No	Job is submitted to the queue that was specified by the submitter.
Disabled (set to false)	No	Yes	Placement rules specify the target queue.
Disabled (set to false)	No	No	Job is submitted to the default queue of the scheduler (root.default).
Enabled (set to true)	Yes	Yes	Placement rules specify the target queue.
Enabled (set to true)	Yes	No	Job is submitted to the queue that was specified by the submitter.

Override Queue Mapping	Target queue is specified at job submission?	Placement rules exist?	End result
Enabled (set to true)	No	Yes	Placement rules specify the target queue.
Enabled (set to true)	No	No	Job is submitted to the default queue of the scheduler (root.default).

When placement rules override the target queue defined at job submission, the specified queue can still be taken into account if the specified placement rule policy is used. For more information, see *Placement rule policies*.

Procedure

1. In Cloudera Manager, select YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Go to the Scheduler Configuration tab.
3. Find the Override Queue Mappings property. It is disabled by default.



4. Check the box to enable this feature.



5. Click Save.
6. Provide a description of the change and then click OK.

Results

Placement rules are considered even if a target queue is specified during a job submission.

Managing dynamic queues

Dynamic queues are created automatically during application runtime. They are deleted when the YARN service is restarted.

Dynamic queues are created during runtime automatically. Dynamic queues are not defined in the capacity-scheduler.xml configuration file. Dynamic queues can be created under those static parents where auto dynamic child creation is allowed. Auto dynamic child creation has to be set explicitly using the YARN Queue Manager UI.

You can create dynamic queues in two ways:

- A dynamic queue path is specified by the submitter at the time of the job submission. Dynamic queues will be created if the auto dynamic child creation feature is enabled for the parent queue that was provided in the queue path.
- A placement rule applies to the submitted job that could place it into a dynamic queue. Dynamic queue is created based on the predefined expression of the dynamic placement rule.

Based on your resource allocation mode, dynamic queues are managed differently:

In absolute and relative modes, when you enable the auto dynamic child creation feature for a queue, it becomes a Managed Parent Queue. It cannot have static child queues, queues under it can be created only dynamically. It allows 1-level dynamic queue nesting.

In the weight mode, there is no Managed Parent Queue. When you enable the auto dynamic child creation feature for a queue, it becomes a parent queue that can have both static and dynamic child queues. It allows 2-level dynamic queue nesting.

Note: Although it is possible to use safety valve configuration snippets to configure dynamic queues, Cloudera recommends to use the YARN Queue Manager UI for dynamic queue configuration even if that leads to some limitations.

Managed Parent Queues

Managed Parent Queues are auto dynamic child creation enabled queues in absolute and relative resource allocation mode.

In absolute and relative modes, when you enable auto dynamic child creation for a queue it becomes a Managed Parent Queue. It cannot have static child queues, and queues under it can be created only dynamically.

In absolute and relative modes, dynamically created queues always fall under a predefined (static) queue, which is the Managed Parent Queue. This limits the nesting to only one level. In addition, the queue properties set for the Managed Parent Queue will be applied to all of its dynamically created child queues. To change the queue properties of all its dynamic child queues, you have to change the configuration at the Managed Parent Queue level.

It is possible to dynamically create queues with zero capacity by incorrectly setting the Managed Parent Queue.

For example, you can create a Managed Parent Queue and assign a percentage based minimum capacity limit of 5%, to its dynamically created child queues. In this case, at most 20 queues can function with the 5% capacity limit. This forces all the following queues to wait until a queue is released (if no application is running in a queue, its capacity is set to zero). Therefore, it is vital to design the minimum capacity limit of child queues that belong to a Managed Parent Queue in a way that takes into account the number of queues that should be running in parallel.

Converting a queue to a Managed Parent Queue

In absolute and relative modes, you have to create Managed Parent Queues to enable dynamic queue creation. You can do that through the YARN Queue Manager UI.

About this task

Note that the queue properties that is set at the Managed Parent Queue level is applied to all of its leaf queues.

Important: Once dynamic child creation is enabled for a queue you cannot disable it. If you enable it by mistake you have to delete the queue and its child queues and then recreate them.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Find the queue for which you want to enable the dynamic child creation feature.
3. Select the More Options menu and select Enable Dynamic Child Creation.



4. Set the Minimum and Maximum capacities that will be applied for every dynamic child queue under that particular parent queue.

   Save the minimum and maximum capacity for the queue with the default values that copy the parent queue, or change them using the Auto Queue Creation template (Technical Preview) feature.

   Technical Preview: This is a technical preview feature and considered under development. Do not use this in your production systems. To share your feedback, contact Support by logging a case on our Cloudera Support Portal. Technical preview features are not guaranteed troubleshooting guidance and fixes.
5. Click Save.

What to do next

To define a placement rule that could lead to dynamically created child queues, ensure that during placement rule creation you select the Create the target queue if it does not exist? option and provide a Managed Parent Queue as the parent queue. For more information, see *Manage placement rules*.

Related Information

Managing placement rules

Enabling dynamic child creation in weight mode

In the weight resource allocation mode, when you enable dynamic child creation for a queue, it becomes a parent queue that can have both static and dynamic child queues. You can enable that feature through the YARN Queue Manager UI.

About this task

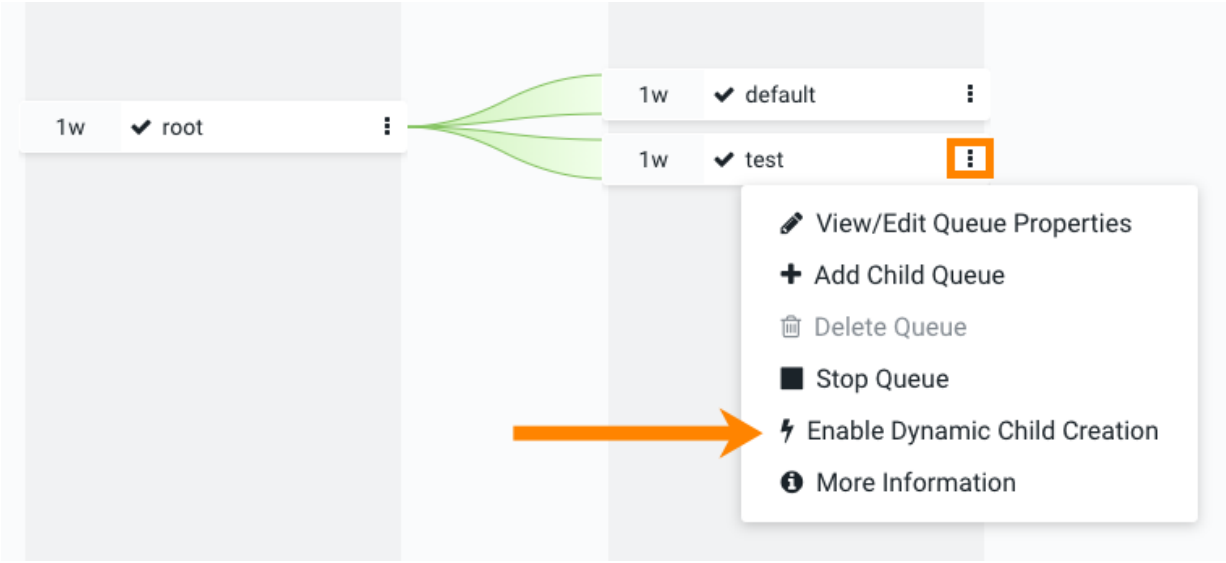
In the weight mode, there is no Managed Parent Queue. When you enable the dynamic child creation feature for a queue, it becomes a parent queue that can have both static and dynamic child queues simultaneously. If this feature is not enabled, the queue can only have static child queues.

In contrast to Manage Parent Queue where the dynamic queue nesting level is limited to one, in weight mode, auto dynamic child creation allows you to create 2-level dynamic queues.

By default dynamic child queues are deleted automatically 5 minutes after the last job finished on them. You can disable this feature using the Capacity Scheduler Auto Queue Deletion YARN property. For more information, see *Disabling Auto Queue Deletion*.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Find the queue for which you want to enable the auto dynamic child creation feature.
3. Select the More Options menu and select Enable Dynamic Child Creation.



Dynamic Child Queue Capacities window is displayed.

4. Set the Minimum and Maximum capacities that will be applied for every dynamic child queue under that particular parent queue.

   Save the minimum and maximum capacity with the default values that copy the parent queue, or change them using the Auto Queue Creation template (Technical Preview) feature.

   Technical Preview: This is a technical preview feature and considered under development. Do not use this in your production systems. To share your feedback, contact Support by logging a case on our Cloudera Support Portal. Technical preview features are not guaranteed troubleshooting guidance and fixes.

Results

Dynamic child creation is enabled for the queue and a bolt icon is visible next to the queue's name.

What to do next

If you want to define a placement rule that could lead to dynamically created child queues, ensure that during placement rule creation you check the Create the target queue if it does not exist? property and provide a parent queue for which dynamic child creation is enabled. For more information, see *Manage placement rules*.

Related Information

Managing placement rules

Placement rule policies

Disabling auto queue deletion globally

Disabling dynamic child creation in weight mode

In the weight resource allocation a dynamic child creation enabled parent queue can have both static and dynamic child queues. If you disable the dynamic child creation feature for a parent queue its dynamic child queues are deleted, and it can have only static queues.

Before you begin

Ensure that no placement rule uses the dynamic child creation enabled parent queue for which you want to disable the dynamic child creation feature. The YARN Queue Manager UI will not let you disable the feature for a parent queue that is used by a placement rule.

Procedure

1. In Cloudera Manager navigate to Clusters YARN Queue Manager UI
2. Find the dynamic child creation enabled queue for which you want to disable the dynamic child creation feature.
3. Click the three vertical dots next to the queue name.
4. Select Disable Dynamic Child Creation.
5. Click Yes to confirm to disable.

   On the YARN Queue Manager UI, and also on the YARN Web UI, the dynamic child queues of the selected parent queue are removed. However, if there were any running applications on such child queues they will continue running in their original queue.
6. Restart the YARN service.

Results

All dynamically created child queues of the selected parent queue are deleted. After the YARN service restart placement rules are reevaluated and applications that were running on the deleted dynamic child queues are either moved to a different queue or are rejected according to the placement rules.

Managing dynamic child creation enabled parent queues

The YARN Queue Manager UI in Cloudera Manager provides an overview of your queue hierarchy where you can view and manage the parent queues for which dynamic child creation is enabled.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager UI.
   A graphical queue hierarchy is displayed in the Overview tab. A bolt icon is displayed next to the queue name of parent queues for which dynamic child creation is enabled.
2. Select the three vertical dots and perform one of the following action for a dynamic child creation enabled parent queue:
   - Edit Queue Properties: In relative and absolute resource allocation modes, you can view and edit the Dynamic Auto-Creation of Queue section of the Queue Properties. In weight resource allocation mode, you can only view that section.
   - Add Child Queue: Supported only in weight resource allocation mode where a dynamic parent queue can have both static and dynamic child queues simultaneously.
   - Delete Queue: You have to stop the queue first before you can delete it.

     Note: You cannot delete the root and the root.default queues.
   - Stop Queue / Stop Queue and Its Children: Stop the queue and its child queues, if there is any.

Note: In weight resource allocation mode, if the queue has dynamically created child queues, you cannot restart the dynamic child queues once you have stopped them.

- Edit Dynamic Child Queue Capacities: Edit the capacities of dynamic child queues created under the parent queue.
- More Information: You are redirected to the Resource Manager UIv2 Queues page.

Managing dynamically created child queues

YARN Queue Manager UI provides an overview of your queue hierarchy where you can view and manage your dynamically created child queues.

About this task

You cannot directly delete dynamically created child queues. For more information about dynamic queue deletion, see *Delete dynamically created child queues*.

Procedure

1. In Cloudera Manager, select the YARN Queue Manager UI.

   A graphical queue hierarchy is displayed in the Overview tab. A leaf is displayed next to the queue name of parent queues for which dynamic child creation is enabled.
2. Select the More Options menu and perform one of the following actions for a dynamically created queue:
   - Edit Queue Properties: Available in relative and absolute resource allocation mode.
   - More Information: You are redirected to the Resource Manager UIv2 Queue page.

Related Information

Deleting dynamically created child queues

Deleting dynamically created child queues

Dynamically created child queues can be deleted automatically once they become inactive, or manually with some limitations.

Automatic child queue deletion

The automatic queue deletion feature is enabled by default. As a result, dynamically created child queues are deleted 300 seconds after their last job is finished. You can disable this feature globally for all dynamic child creation enabled parent queues or for specific parent queues. Additionally, configuring the queue removal expiration time is also possible.

Note: This feature is called auto queue deletion on the global level and queue auto removal on the queue level.

Manual child queue deletion

Unlike parent queues, dynamically created child queues cannot be deleted manually. No action or button exists that deletes a specific dynamically created child queue. Instead, you can delete them by either restarting the YARN service or deleting the parent queue.

Disabling auto queue deletion globally

Dynamically created child queues are automatically deleted after they expire. If you want to keep all dynamically created child queues running after their last job is finished, you can disable automatic queue deletion on a global level.

About this task

By default automatic queue removal is enabled on two levels: globally and on the level of dynamic child creation enabled parent queues. The following steps describe how to disable the auto queue deletion feature on a global level, that is, dynamically created child queues are not deleted automatically once they become inactive.

Note: If you want to disable the automatic queue removal for specific dynamic child creation enabled parent queues, use the Queue Auto Removal (YARN restart required) Queue Manager property. For more information, see Disabling queue auto removal on a queue level.

Procedure

1. In Cloudera Manager, navigate to YARN Configuration .
2. Search for queue deletion.
3. Find and clear the Capacity Scheduler Auto Queue Deletion property to disable the auto queue deletion feature.
4. Click Save.
5. Restart the YARN service.

Results

Auto queue deletion is disabled. Dynamically created queues do not get deleted once they became inactive.

Related Information

Disabling queue auto removal on a queue level

Disabling queue auto removal on a queue level

Dynamically created child queues are automatically deleted after they expire. If you want to keep dynamically created child queues of a specific parent queue running after their last job is finished, you can disable the queue auto removal feature for that parent queue.

About this task

By default automatic queue removal is enabled on two levels - global level and on the level of dynamic child creation enabled parent queues. The following steps describe how to disable the queue auto removal feature for a dynamic child creation enabled parent queue; meaning, its dynamically created child queues are not deleted automatically once they become inactive.

Note: If you want to disable the automatic queue removal for all dynamically created child queues, you can use the global Capacity Scheduler Auto Queue Deletion YARN property. For more information, see *Disabling auto queue deletion globally*.

Before you begin

Ensure that the global level auto queue deletion feature is enabled. The feature is enabled if the YARN Configuration Capacity Scheduler Auto Queue Deletion checkbox is selected.

Procedure

1. In Cloudera Manager navigate to Clusters YARN Queue Manager UI .
   The Overview tab is opened, displaying a hierarchical view of the queues.
2. Disable the queue auto removal feature for a specific queue.

   Repeat the following substeps for all applicable dynamic child creation enabled queues:

a) Find the queue for which you want to disable the queue auto removal feature, and click the three vertical dots

next to its name.
b) Select Edit Queue Properties.
c) In the Dynamic Auto-Creation of Queue section find and clear the Queue Auto Removal (YARN restart required) checkbox.
d) Click Save.

Results

Auto queue deletion is enabled on the global level and disable for the selected dynamic child creation enabled queues.

Related Information

Disabling auto queue deletion globally

Configuring the queue auto removal expiration time

The expiration time for automatic queue deletion is set to 300 seconds by default. You can change this value using the YARN Queue Manager UI.

Procedure

1. In Cloudera Manager navigate to Clusters YARN Queue Manager UI .
   The Overview tab is opened, displaying a hierarchical view of the queues.
2. Click Configuration.
3. Find the Queue Auto Removal Expiration Time property and set the expiration time in seconds.

   The default expiration time is 300, meaning that if auto queue deletion is enabled for a queue, its dynamically created child queues are deleted 300 seconds after the last job finished on them.

   
   Note: Automatic queue deletion is supported in the weight mode only. This feature is not supported in relative or absolute resource allocation modes.
4. Click Save.
5. Restart the YARN service.

Deleting dynamically created child queues manually

Deleting dynamically created child queues manually is not supported by the YARN Queue Manager UI, but there are some workarounds to remove them. It can be useful, for example, when the applications in that queue are terminated.

There are two ways to remove dynamically created child queues manually:

- Restart the YARN service: That stops and deletes all dynamically created queues.
- Stop and then delete the parent queue of the dynamically created child queues: This will delete both the dynamic parent and all of its child queues - both static and dynamic ones.

Partition configuration

You can partition a cluster into sub-clusters so that jobs run on nodes with specific characteristics. You can configure these partitions, so that you run YARN applications on cluster nodes of the specified partition.


Note: The term *Partitions* is used instead of *Node Labels* to be consistent with the YARN terminology.

Partition can be specified as exclusive or non-exclusive/shareable

- exclusive - Access is restricted to applications running in queues associated with the partition.

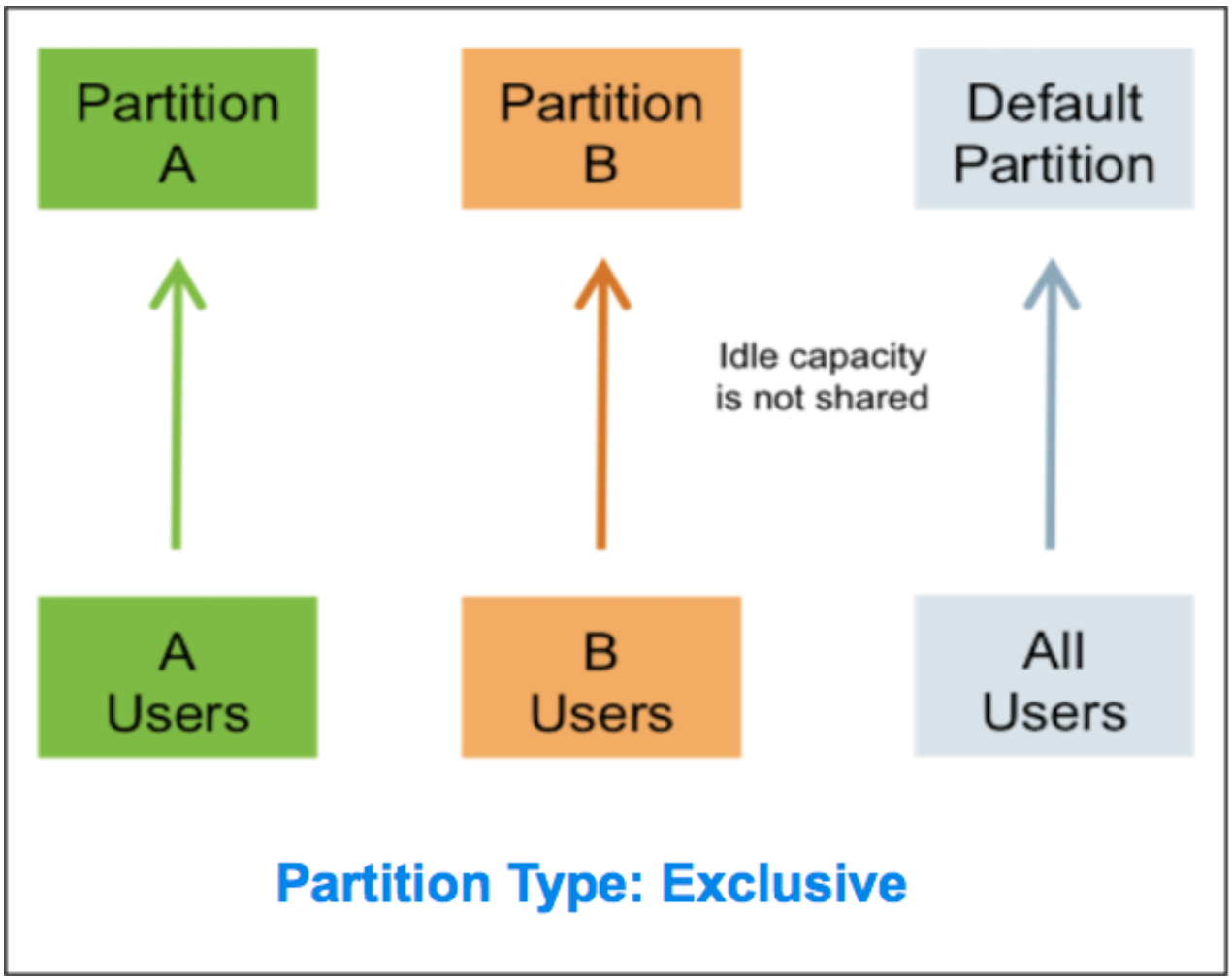
- non-exclusive - If idle capacity is available on the partition, resources are shared with all applications in the cluster.

The fundamental unit of scheduling in YARN is the queue. The capacity of each queue specifies the percentage of cluster resources that are available for applications submitted to the queue. Queues can be set up in a hierarchy that reflects the resource requirements and access restrictions required by the various organizations, groups, and users that utilize cluster resources.

Using partition, you can divide a cluster into sub-clusters so that jobs can be run on partitions with specific characteristics. For example, you can use a partition to run memory-intensive jobs only on nodes with a larger amount of RAM. Partitions can be assigned to cluster nodes, and specified as exclusive or non-exclusive. You can then associate partition with capacity scheduler queues. Each node can be associated with only one partition.

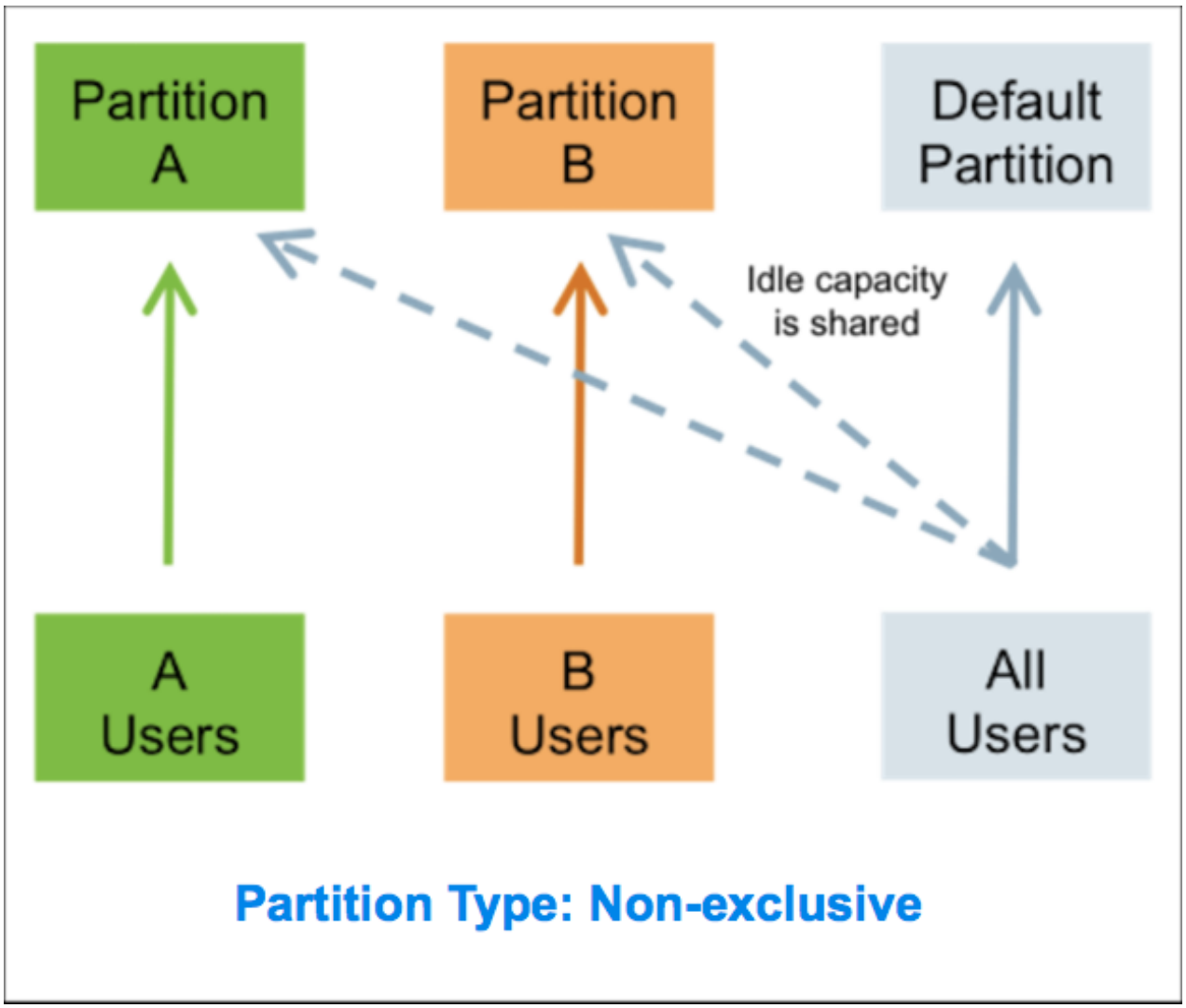
Partition Type: Exclusive

When a queue is associated with one or more exclusive partitions, all applications submitted by the queue have exclusive access to the nodes in those partitions.



Partition Type: Non-exclusive

When a queue is associated with one or more non-exclusive partitions, all applications submitted by the queue get first priority on nodes in those partitions. If idle capacity is available on those partition nodes, resources are shared with other applications in the cluster. Non-labeled applications are preempted if labeled applications request new resources on the labeled nodes.



Queues without associate partition

If no partition is assigned to a queue, the applications submitted by the queue can run on any node without a partition, and on nodes with non-exclusive partitions if idle resources are available.

Preemption

Labeled applications that request labeled resources preempt non-labeled applications on labeled nodes. If a labeled resource is not explicitly requested, the normal rules of preemption apply. Non-labeled applications cannot preempt labeled applications running on labeled nodes.

Enabling node labels on a cluster to configure partition

You can configure partitions on a cluster by making configuration changes on the YARN ResourceManager host.

Procedure

1. Create a Label Directory in HDFS.
   a) Use the following commands to create a node-labels directory in which to store the Node Labels in HDFS:

```
sudo su hdfs
```

```
hadoop fs -mkdir -p /yarn/node-labels
hadoop fs -chown -R yarn:yarn /yarn
hadoop fs -chmod -R 700 /yarn
```

Where -chmod -R 700 specifies that only the yarn user can access the node-labels directory.

b) Use the following command to confirm that the directory was created in HDFS:

```
hadoop fs -ls /yarn
```

The new node-labels directory should appear. The owner should be yarn, and the permission should be drwx:

```
Found 1 items
drwx------ - yarn yarn 0 2014-11-24 13:09 /yarn/node-labels
```



Note: In a HA HDFS environment, ensure to always use a HDFS name service and not just a NameNode.

2. In Cloudera Manager, navigate to YARN Configuration .
3. Search for node label.
4. Find the Node Labels property and ensure that it is selected (enabled).
5. Find the Node Labels Directory property and set its value to reference the HDFS node label directory.

   For example: hdfs://node-1.example.com:8020/yarn/node-labels/.

   HDFS paths are automatically translated, however, any other file system requires a full path to the Node Labels Directory.

   The default value of the Node Labels Directory property is /yarn/node-labels.
6. Start or Restart the YARN ResourceManager.
7. Restart YARN Queue Manager.
8. Refresh the Yarn QueueManager UI.

Creating partitions

You must first create partitions to assign them to nodes and associate it with queues.

Before you begin

You must enable node labels on a cluster before you create a partition. For more information, see Enable node labels on a cluster.

Procedure

1. In Cloudera Manager, navigate to Clusters YARN Queue Manager UI service.
   A graphical queue hierarchy is displayed in the Overview tab.
2. Click Partitions.
3. Click +Create.
   The Create Partition dialog box is displayed.
4. Add a name for the partition in Partition Name.
5. Select Exclusive or Non-Exclusive node label type under Partition Type.

   For information about Exclusive or Non-Exclusive partition type, see Configure Partitions.
6. Select one or more unassigned nodes listed under Unassigned Nodes and click the < arrow to move it under Assigned Nodes to assign it to the partition. You can also search or filter the nodes using the regular expression.
7. Click Save.

Assigning or unassigning a node to a partition

You can assign or unassign a node to an existing partition.

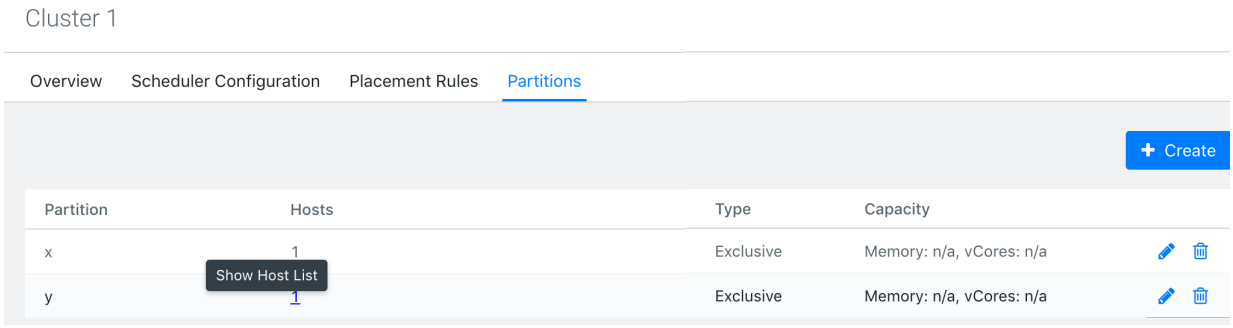
Procedure

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Partition tab. A list of partitions is displayed.
3. Click the Edit icon at the right of a partition.
4. Assign or unassign the nodes in the Edit Partition dialog box.
   a. Assign Nodes: Select an unassigned node listed under Unassigned Nodes and click the < arrow button to move it to under Assigned Nodes.
   b. Unassign Nodes: Select an assigned node listed under Assigned Nodes and click the > arrow button to move it to under Unassigned Nodes.
5. Click Save.

Viewing partitions

You can view the list of available partitions in the cluster. For each partition, it lists the number of associated nodes under the Hosts column along with the partition type and capacity.

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click the Partitions tab. A list of existing partitions is displayed.
3. Optionally, you can click on the number listed in the Host column to view the associated nodes.



View Node Label Assignments

You can use the following commands to view information about partitions.

- List all running nodes in the cluster: yarn node -list

  Example:

```
[root@node-1 /]# yarn node -list
14/11/21 12:14:06 INFO impl.TimelineClientImpl: Timeline service address:
 http://node-1.example.com:8188/ws/v1/timeline/
14/11/21 12:14:07 INFO client.RMProxy: Connecting to ResourceManager at no
de-1.example.com/240.0.0.10:8032
Total Nodes:3
 Node-Id Node-State Node-Http-Address Number-of-Running-Containers
node-3.example.com:45454 RUNNING node-3.example.com:50060 0
```

```
node-1.example.com:45454 RUNNING node-1.example.com:50060 0
node-2.example.com:45454 RUNNING node-2.example.com:50060 0
```

- List the status of a node (includes partition): yarn node -status <Node_ID>

  Example:

```
[root@node-1 /]# yarn node -status node-1.example.com:45454
14/11/21 06:32:35 INFO impl.TimelineClientImpl: Timeline service address:
 http://node-1.example.com:8188/ws/v1/timeline/
14/11/21 06:32:35 INFO client.RMProxy: Connecting to ResourceManager at
 node-1.example.com/240.0.0.10:8032
Node Report :
 Node-Id : node-1.example.com:45454
 Rack : /default-rack
 Node-State : RUNNING
 Node-Http-Address : node-1.example.com:50060
 Last-Health-Update : Fri 21/Nov/14 06:32:09:473PST
 Health-Report :
 Containers : 0
 Memory-Used : 0MB
 Memory-Capacity : 1408MB
 CPU-Used : 0 vcores
 CPU-Capacity : 8 vcores
 Node-Labels : x
```

Partitions are also displayed in the ResourceManager UI on the Nodes and Scheduler pages.

Associating partitions with queues

You can use partitions to run YARN applications on cluster nodes that have a specified partition.

Before you begin

Before associating partitions, you must create partitions and assign partitions to cluster nodes. For more information about creating partitions see, Creating partitions on page 91.

Note: After you associate a partition with one or more queues, in the YARN Queue Manager UI, click Overview > <*PARTITION NAME*> from the drop-down list and distribute capacity to the queues before switching allocation mode or creating placement rules.

Note: The term *Partitions* is used instead of *Node Labels* to be consistent with the YARN terminology.

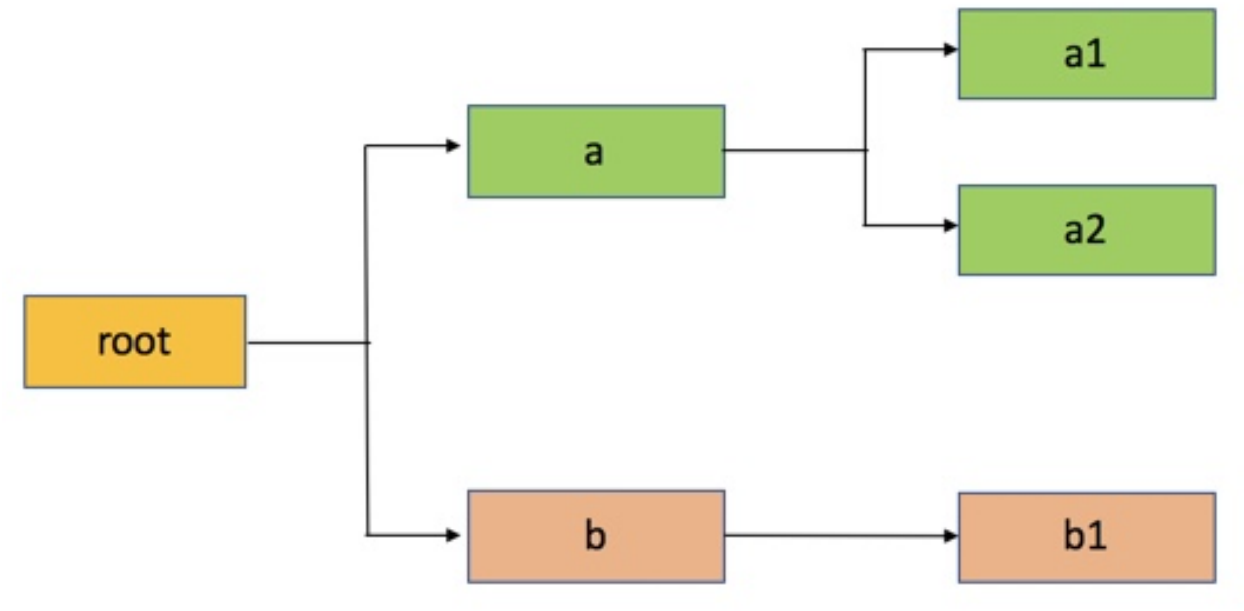
About this task

Use Queue Manager to create and assign partitions to cluster nodes, associate the partitions (yarn.scheduler.capacity.<queue-path>.accessible-node-labels) to queues and configure capacity to that queue for the specified partition. Queue Manager evenly distributes the available capacity among all the queues in the partition. You can manually modify the capacity on each partition of each queue, and also ensure that the sum of capacities of each partition of direct children of a parent queue at every level is equal to 100%. partitions that a queue can access (accessible partitions of a queue) must be the same as, or a subset of, the accessible partitions of its parent queue.

Example

Assume that a cluster has a total of 8 nodes. The first 3 nodes (n1-n3) have partition = x, the next 3 nodes (n4-n6) have partition = y, and the final 2 nodes (n7, n8) do not have any partitions. Each node can run 10 containers.

The queue hierarchy is as follows:



Assume that queue “a” can access partitions “x” and “y”, and queue “b” can only access partition “y”. By definition, nodes without labels can be accessed by all queues.

Consider the following example label configuration for the queues, in the Relative resource allocation mode:

capacity(a) = 40, capacity(a, label=x) = 100, capacity(a, label=y) = 50; capacity(b) = 60, capacity(b, label=y) = 50

This means that:

- Queue “a” can access 40% of the resources on nodes without any labels, 100% of the resources on nodes with label=x, and 50% of the resources on nodes with label=y.
- Queue “b” can access 60% of the resources on nodes without any labels, and 50% of the resources on nodes with label=y.

You can also see that for this configuration:

capacity(a) + capacity(b) = 100

capacity(a, label=x) + capacity(b, label=x) (b cannot access label=x, it is 0) = 100

capacity(a, label=y) + capacity(b, label=y) = 100

For child queues under the same parent queue, the sum of the capacity for each label should equal 100%.

Similarly, you can set the capacities of the child queues a1, a2, and b1:

a1 and a2: capacity(a.a1) = 40, capacity(a.a1, label=x) =30, capacity(a.a1, label=y) =50 capacity(a.a2) = 60, capacity(a.a2, label=x) =70, capacity(a.a2, label=y) =50;

b1: capacity(b.b1) = 100, capacity(b.b1, label=y) = 100

You can see that for the a1 and a2 configuration:

capacity(a.a1) + capacity(a.a2) = 100

capacity(a.a1, label=x) + capacity(a.a2, label=x) = 100

capacity(a.a1, label=y) + capacity(a.a2, label=y) = 100

How many resources can queue a1 access?

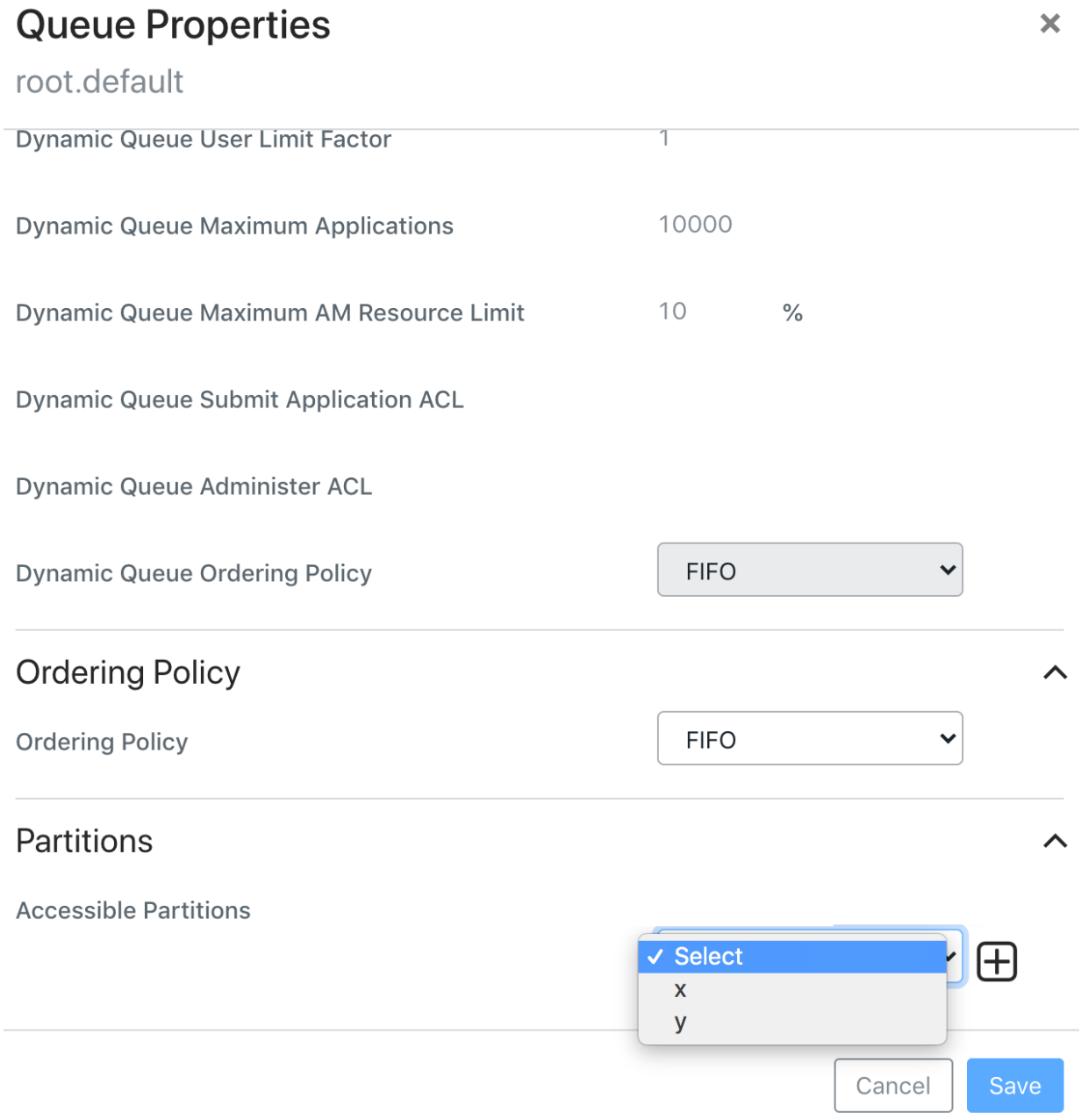
Resources on nodes without any labels: Resource = 20 (total containers that can be allocated on nodes without a label, in this case n7, n8) * 40% (a.capacity) * 40% (a.a1.capacity) = 3.2 (containers)

Resources on nodes with label=x

Resource = 30 (total containers that can be allocated on nodes with label=x, in this case n1-n3) * 100% (a.labelx.capacity) * 30% = 9 (containers)

To implement this example configuration, perform the following

1. In Cloudera Manager, select Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on a queue and select the View/Edit Queue Properties option.
3. In the Queue Properties dialog-box , select the x label from the Accessible Partitions drop-down box, click +, again select the y label from the Accessible Partitions drop-down box, and click Save.



4. Repeat the above steps to assign x label for a1 and a2 queues.
5. Click on the three vertical dots on b queue and select the Edit Queue Properties option.
6. In the Queue Properties dialog-box, select the y label from the Accessible Partitions drop-down box, click +, and click Save.

7. Repeat the above steps to assign y label for b1, a, a1, and a2 queue.

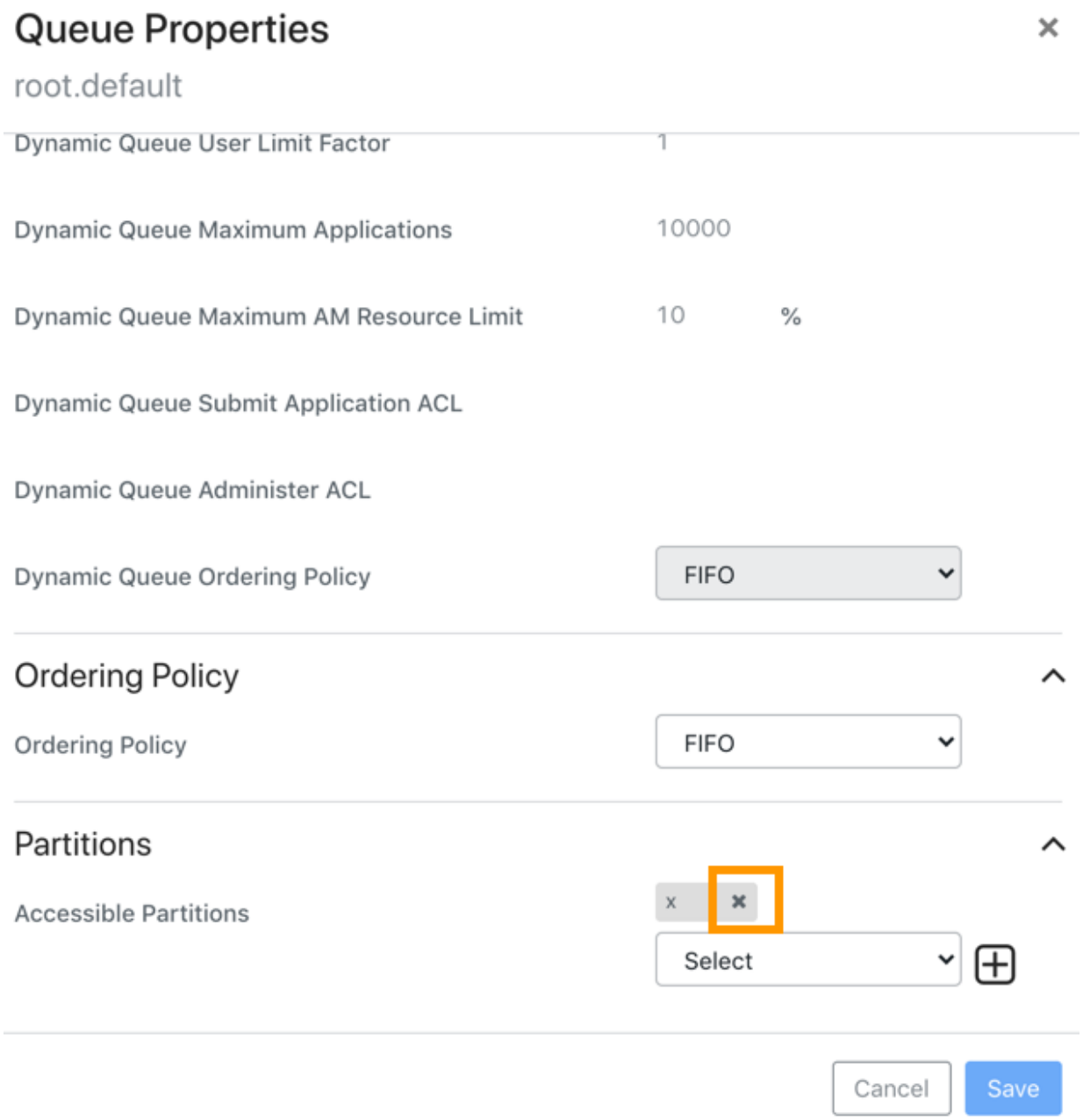
   Queue Manager automatically distributes the available capacity among all the queues in the partition. If you want to modify the capacity of the queues, click on the Partition drop-down box in the Overview tab, select the label and modify the queue capacity.
8. In the Overview tab, click on the Partition drop-down box and select label y.
9. Click on the three vertical dots on the a queue and select the Edit Child Queues option.
10. Enter the Configured Capacity of a1 to 50 and a2 to 50 and Click Save.
11. Click on the three vertical dots on the b queue and select the Edit Child Queues option.
12. Enter the Configured Capacity of b1 to 100 and click Save.
13. Click on the three vertical dots on the root queue and select the Edit Child Queues option.
14. Enter the Configured Capacity of a to 50 and b to 50 and click Save.

Disassociating partitions from queues

You can disassociate a partition from the queue. You should disassociate a partition before you delete the queue. Before disassociating a partition from a queue, you should remove the partition capacity for that queue by setting it to zero.

Procedure

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on a queue and select the View/Edit Queue Properties option.
3. In the Queue Properties dialog box, from the Accessible Partitions, click X next to the name of the partition.



4. Click Save.

Deleting partitions

Due to a known issue, COMPX-5106, it is not recommended to delete a partition if it is associated with queues and the queues have capacities configured for that partition.

Before you begin

You must have created a partition in the cluster using YARN /add-node-labels API, and assigned this partition to some queues and set the partition capacities to queues.

About this task

Use Queue Manager to create a partition in the cluster using YARN /add-node-labels API and assign this partition to some queues and set the partition capacities to queues as follows:

1. Create the partition demo
2. Assign the partition to queues:
   - yarn.scheduler.capacity.root.accessible-node-labels=”*”
   - yarn.scheduler.capacity.root.default.accessible-node-labels=”demo”
3. Set the partition capacities to queues:
   - yarn.scheduler.capacity.root.accessible-node-labels.demo.capacity=100
   - yarn.scheduler.capacity.root.accessible-node-labels.demo.maximum-capacity=100
   - yarn.scheduler.capacity.root.default.accessible-node-labels.demo.capacity=100
   - yarn.scheduler.capacity.root.default.accessible-node-labels.demo.maximum-capacity=100

Now, to delete the partition and its related configs that are created above perform the following:

1. Unassign node labels from the queues set in step-2 above:
   - yarn.scheduler.capacity.root.accessible-node-labels=”*”
   - yarn.scheduler.capacity.root.default.accessible-node-labels=””
2. Delete the partition created in step-1 using the YARN /remove-node-labels API.

Now the system is left with residue configs that was set in step-3. And there is no way to remove these and causes various issues when trying to remove the residue configs.

Solution to remove the configs

YARN’s mutation API is designed to be simple and very generic, it only handles key-value pairs. The unwanted capacities can be removed by using an existing mutation API in YARN. This is performed by setting values to an empty (“”) string (or null) and then those configs will be removed from the in-memory config.

For example:

```
{
    "update-queue":
    [
        {
            "queue-name": "root",
            "params":
            {
                "entry":
                [
                    {
                        "key": "accessible-node-labels.label1.maximum-capac
ity",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels",
                        "value": "*"
                    },
                    {
                        "key": "accessible-node-labels.label1.capacity",
                        "value": ""
                    }
                ]
            }
        },
{
            "queue-name": "root.a",
            "params":
```

```
            {
                "entry":
                [
                    {
                        "key": "accessible-node-labels.label1.maximum-cap
acity",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels.label1.capacity",
                        "value": ""
                    }
                ]
            }
        },
        {
            "queue-name": "root.a.a1",
            "params":
            {
                "entry":
                [
                    {
                        "key": "accessible-node-labels.label1.maximum-cap
acity",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels.label1.capacity",
                        "value": ""
                    }
                ]
            }
        },
        {
            "queue-name": "root.a.a2",
            "params":
            {
"entry":
                [
                    {
                        "key": "accessible-node-labels.label1.maximum-cap
acity",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels",
                        "value": ""
                    },
                    {
                        "key": "accessible-node-labels.label1.capacity",
                        "value": ""
                    }
                ]
            }
        }
    ],
```

```
    "global": null,
    "global-updates": null
}
```

```
{
    "key": "accessible-node-labels.label1.capacity",
    "value": ""
}
```

and

```
{
    "key": "accessible-node-labels.label1.maximum-capacity",
    "value": ""
}
```

As seen in the above JSON, add null or empty string values for all the queues where you want to clear values from the config.

Setting a default partition expression

You can set a default partition on a queue. The default partition is used if no partition is specified when the application is submitted to the queue. By default, this field is empty, so applications get containers from nodes without a partition or label.

Procedure

1. In Cloudera Manager, select the Clusters > YARN Queue Manager UI service. A graphical queue hierarchy is displayed in the Overview tab.
2. Click on the three vertical dots on the queue and select the View/Edit Queue Properties option.
3. In the Queue Properties dialog-box, select the default partition from the Default Partition Expression drop-down list.
4. Click Save.

Using partitions when submitting a job

You can use various methods to specify partitions when submitting jobs.

Procedure

- Set partitions when Submitting Jobs

  You can use the following methods to specify partitions when submitting jobs:

  - ApplicationSubmissionContext.setNodeLabelExpression(<node_label_expression>) - sets the partition expression for all containers of the application.
  - ResourceRequest.setNodeLabelExpression(<node_label_expression>) - sets the partition expression for individual resource requests. This overrides the partitions expression set in ApplicationSubmissionContext.set NodeLabelExpression(<node_label_expression>).
  - Specify setAMContainerResourceRequest.setNodeLabelExpression in ApplicationSubmissionContext to indicate the expected partition for the ApplicationMaster container.

  You can use one of these methods to specify a partition expression, and -queue to specify a queue, when you submit YARN jobs using the distributed shell client. If the queue has a label that satisfies the label expression, it will run the job on the partition(s). If the label expression does not reference a label associated with the specified

queue, the job does not run and an error is returned. If no partitionn is specified, the job runs only on nodes without a partition, and on nodes with non-exclusive partitions if idle resources are available.



Note:

You can only specify one partition in the .setNodeLabelExpression methods.

For example, the following commands run a simple YARN distributed shell "sleep for a long time" job. In this example you are asking for more containers than the cluster can run so you can see which node the job runs on. We are specifying that the job should run on queue "a1", which our user has permission to run jobs on. We are also using the -node_label_expression parameter to specify that the job will run on all nodes with label "x".

```
sudo su yarn
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-yarn/hadoop-yarn-applic
ations-distributedshell.jar
 -shell_command "sleep 100" -jar /opt/cloudera/parcels/CDH/lib/hadoop-ya
rn/hadoop-yarn-applications-distributedshell.jar
 -num_containers 30 -queue a1 -node_label_expression x
```

If you run this job on the example cluster we configured previously, containers are allocated on node-1, as this node has been assigned partition "x", and queue "a1" also has partition "x":

The following commands run the same job that you specified for partition "x", but this time you will specify queue "b1" rather than queue "a1".

```
sudo su yarn
hadoop jar /opt/cloudera/parcels/CDH/lib/hadoop-yarn/hadoop-yarn-applic
ations-distributedshell.jar
 -shell_command "sleep 100000" -jar /opt/cloudera/parcels/CDH/lib/hadoop-
yarn/hadoop-yarn-applications-distributedshell.jar
 -num_containers 30 -queue b1 -node_label_expression x
```

When you attempt to run this job on our example cluster, the job will fail with the following error message because label "x" is not associated with queue "b1".

```
14/11/24 13:42:21 INFO distributedshell.Client: Submitting application to
 ASM
14/11/24 13:42:21 FATAL distributedshell.Client: Error running Client
org.apache.hadoop.yarn.exceptions.InvalidResourceRequestException: Invalid
 resource request, queue=b1 doesn't
have permission to access all labels in resource request. labelExpression
 of resource request=x. Queue labels=y
```

- MapReduce Jobs and Partitions

  Currently you cannot specify a partition when submitting a MapReduce job. However, if you submit a MapReduce job to a queue that has a default partition expression, the default partition is applied to the MapReduce job.

  Using default partition expressions tends to constrain larger portions of the cluster, which at some point starts to become counter-productive for jobs - such as MapReduce jobs - that benefit from the advantages offered by distributed parallel processing.