

CDP One

Monitor & Operate

Date published: 2022-06-03

Date modified: 2022-08-15

CLOUDBERA

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2024. 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

| | |
|---|---------------|
| Monitoring workloads..... | 4 |
| Monitoring YARN Applications..... | 6 |
| Viewing Jobs..... | 6 |
| Configuring YARN Application Monitoring..... | 7 |
| Results Tab..... | 7 |
| Filtering Jobs..... | 8 |
| Filter Expressions..... | 8 |
| Choosing and Running a Filter..... | 9 |
| Filter Attributes..... | 11 |
| Sending Diagnostic Data to Cloudera for YARN Applications..... | 18 |
| Monitoring Impala Queries..... | 19 |
| Viewing Queries..... | 19 |
| Configuring Impala Query Monitoring..... | 19 |
| Impala Best Practices..... | 20 |
| Results Tab..... | 20 |
| Filtering Queries..... | 21 |
| Filter Expressions..... | 21 |
| Filter Attributes..... | 21 |
| Choosing and Running a Filter..... | 26 |
| Analyzing YARN jobs..... | 28 |
| Viewing the Cluster Overview..... | 30 |
| Viewing nodes and node details..... | 32 |
| Viewing queues and queue details..... | 33 |
| Viewing all applications..... | 35 |
| Searching applications..... | 36 |
| Viewing application details..... | 37 |
| UI Tools..... | 38 |
| Using the YARN CLI to viewlogs for applications..... | 38 |
| Analyzing Spark jobs..... | 40 |
| Accessing the Spark History Server web UI..... | 40 |
| Viewing YARN queues..... | 47 |
| Adding queues using YARN Queue Manager UI..... | 49 |
| Configuring cluster capacity with queues..... | 54 |
| Configuring the resource capacity of root queue in absolute mode..... | 55 |
| Changing resource allocation mode..... | 58 |
| Starting and stopping queues..... | 59 |
| Deleting queues..... | 60 |
| Setting queue priorities..... | 60 |
| Viewing the YARN job history..... | 61 |

Monitoring workloads

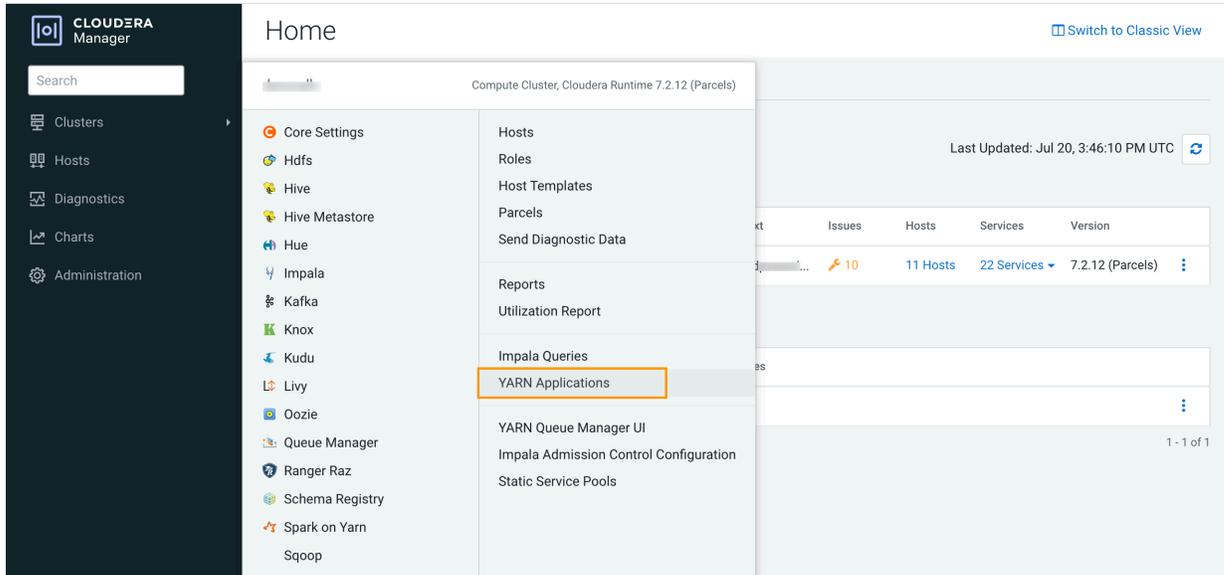
You can use Cloudera Manager to monitor YARN and Impala workloads on CDP One.

Procedure

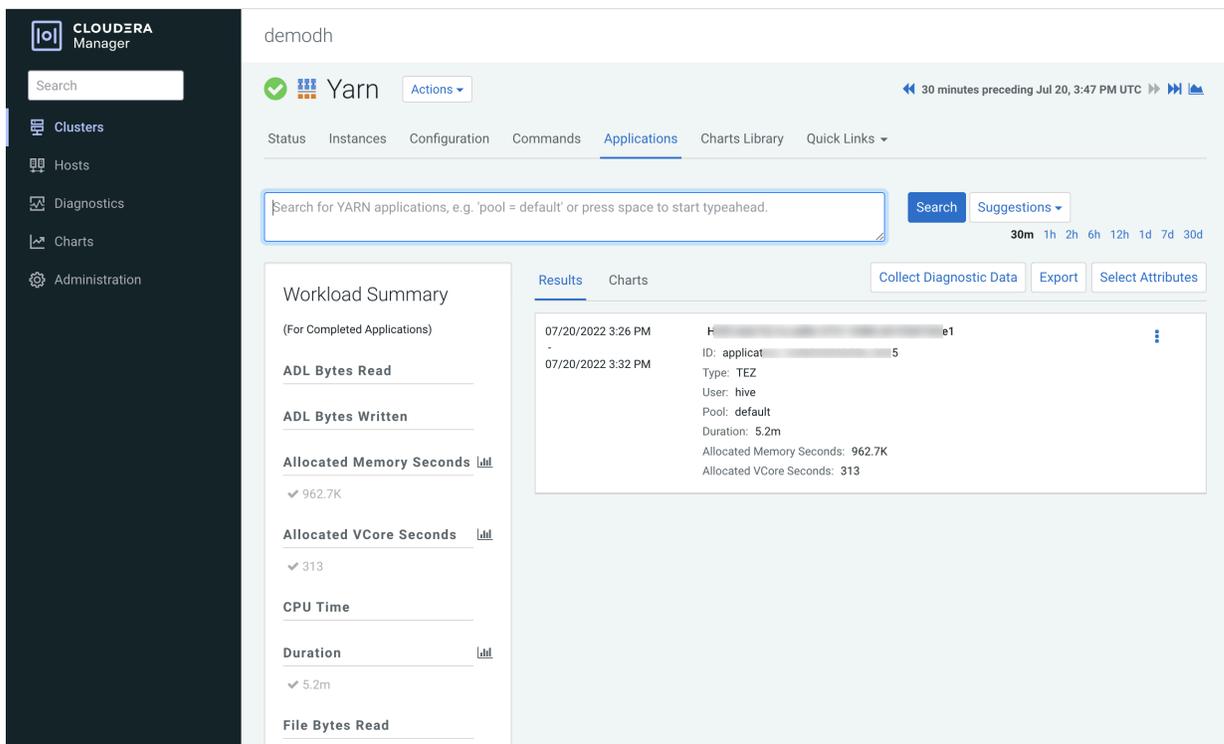
1. Click Monitor Workload on the CDP One console, then select a cluster.

The screenshot displays the Cloudera Data Platform (CDP) console interface. At the top, it says "WELCOME, [User]. What would you like to work on?". Below this are four main navigation buttons: INGEST, PREPARE, ANALYZE, and PUBLISH. The main content area is divided into three columns: Ingest, Query & Notebooks, and Monitor & Operate. In the Monitor & Operate column, the "Monitor Workload with Cloudera Manager" option is highlighted. To the right of the main console, a "Monitor Workload" dialog box is open, showing a "Select Cluster" section with three cluster selection options, each with a dropdown arrow icon.

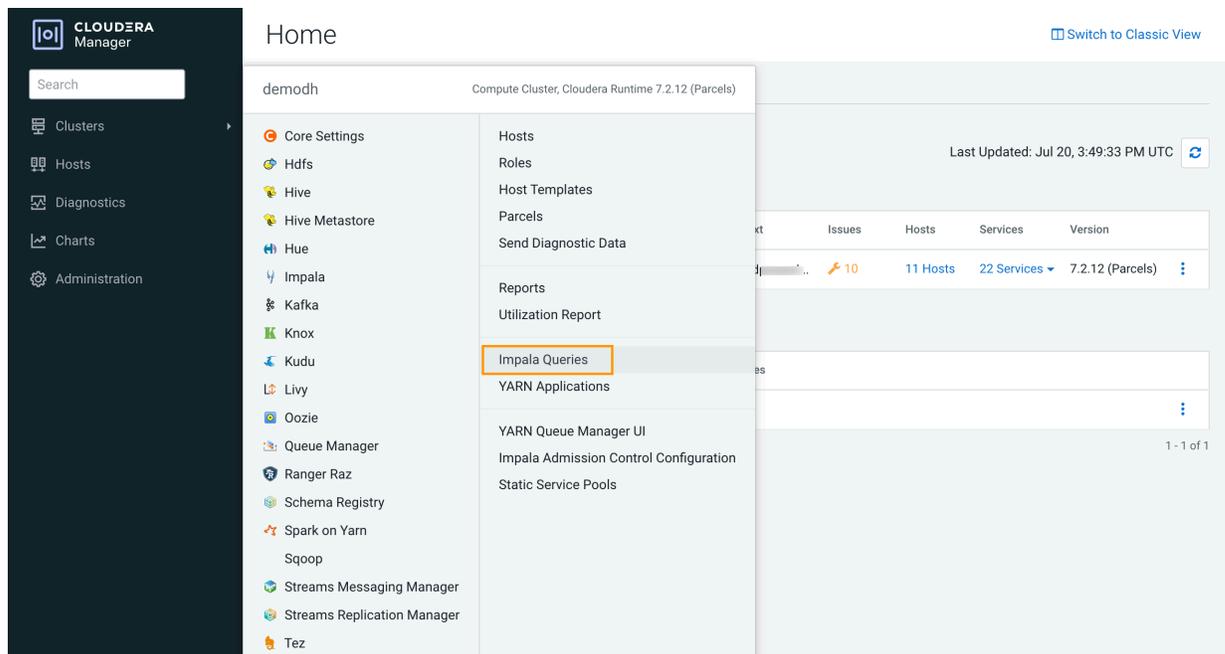
2. To monitor YARN applications, click Clusters, then click YARN Applications.



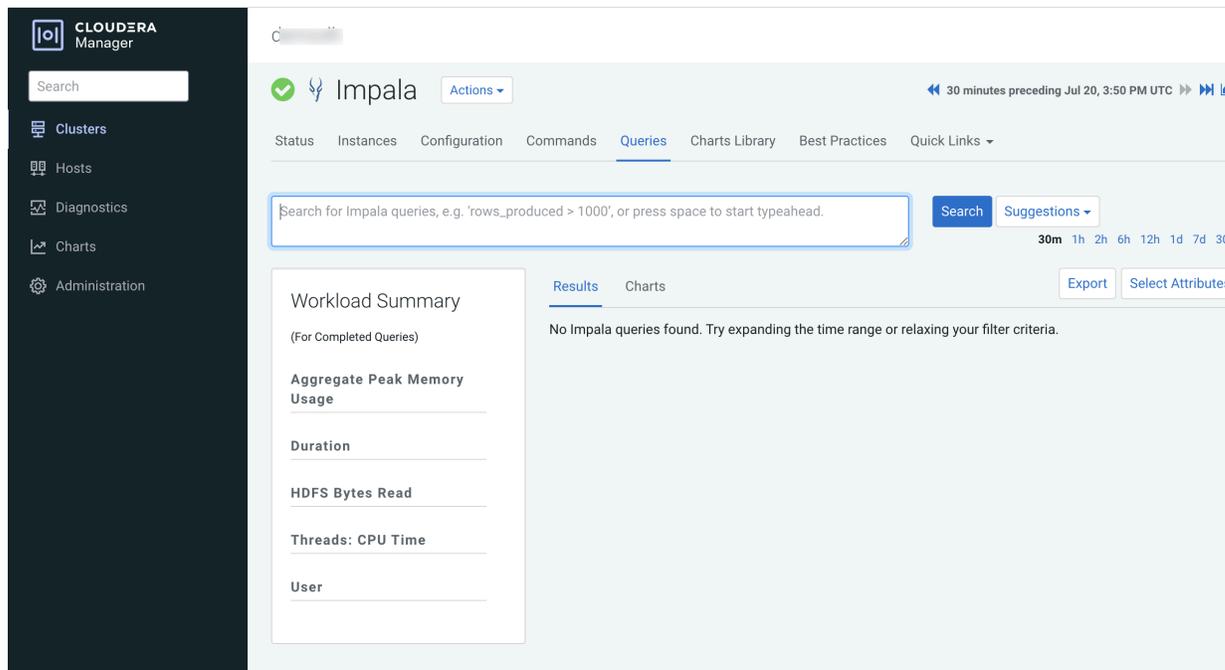
The YARN Applications page appears:



3. To monitor Impala queries, click Clusters, then click Impala Queries.



The Impala Queries page appears:



Monitoring YARN Applications

The YARN Applications page displays information about the YARN jobs that are running and have run in your cluster. You can filter the jobs by time period and by specifying simple filtering expressions.

Viewing Jobs

You can view YARN jobs, filter YARN jobs, and more from the YARN service **Applications** tab.

Do one of the following:

- Select Clusters *Cluster name* *YARN service name* Applications .
- On the Home Status tab, select *YARN service name* and click the Applications tab.

The YARN jobs run during the selected time range display in the Results tab. The results displayed can be filtered by creating filter expressions.

You can also perform the following actions on this page:

Table 1: Viewing Jobs Actions

| Action | Description |
|--|--|
| Filter jobs that display. | Create filter expressions manually, select preconfigured filters, or use the Workload Summary section to build a query interactively. |
| Select additional attributes for display. | Click Select Attributes. Selected attributes also display as available filters in the Workload Summary section. To display information about attributes, hover over a field label. Only attributes that support filtering appear in the Workload Summary section. |
| View a histogram of the attribute values. | Click the  icon to the right of each attribute displayed in the Workload Summary section. |
| Display charts based on the filter expression and selected attributes. | Click the Charts tab. |
| Send a YARN application diagnostic bundle to Cloudera support. | Click Collect Diagnostics Data. |
| Export a JSON file with the query results that you can use for further analysis. | Click Export. |

Configuring YARN Application Monitoring

You can configure the visibility of the YARN application monitoring results.

About this task

To configure whether admin and non-admin users can view all applications, only that user's applications, or no applications:

Minimum Required Role: [Configurator](#) (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

1. Go to the YARN service.
2. Click the Configuration tab.
3. Select Scope *YARN service_name* (Service-Wide) .
4. Click the Monitoring category.
5. Set the Applications List Visibility Settings properties for admin and non-admin users.
6. Enter a Reason for change, and then click Save Changes to commit the changes.
7. Click the Cloudera Manager logo to return to the Home page.
8. Click the  icon that is next to any stale services to invoke the cluster restart wizard.

Results Tab

Jobs appear on the Results tab, with the most recent at the top. Each job has summary and detail information.

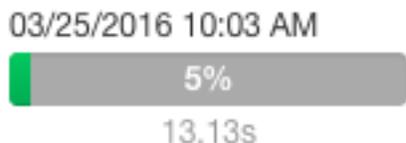
Jobs are ordered with the most recent at the top. Each job has summary and detail information. A job summary includes start and end timestamps, query (if the job is part of a Hive query) name, pool, job type, job ID, and user. For example:

```

03/11/2016 5:30 PM - insert into traffic_lights_complex...street2(Stage-1)
03/11/2016 5:30 PM Hive Query String: > insert into traffic_lights_complex select id, street1, street2, collect_list(named_struct('incident_i...
ID: job\_1455752426632\_0029 Type: MAPREDUCE
User: foo Pool: root.foo
Duration: 14.53s CPU Time: 4.3s
File Bytes Read: 144 B File Bytes Written: 465.6 KIB
HDFS Bytes Read: 22.7 KIB HDFS Bytes Written: 1.7 KIB
Memory Allocation: 9.3M

```

A running job displays a progress bar under the start timestamp:



Use the Actions drop-down menu to the right of each job listing to do the following. (Not all options display, depending on the type of job.)

- Application Details – Open a details page for the job.
- Collect Diagnostic Data – Send a YARN application diagnostic bundle to Cloudera support.
- Similar MR2 Jobs – Display a list of similar MapReduce 2 jobs.
- User's YARN Applications – Display a list of all jobs run by the user of the current job.
- View on JobHistory Server – View the application in the YARN JobHistory Server.
- Kill (running jobs only) – Stop a job (administrators only). Stopping a job creates an audit event. When you stop a job, **Killed** replaces the progress bar.
- Applications in Hive Query (Hive jobs only)
- Applications in Oozie Workflow (Oozie jobs only)
- Applications in Pig Script (Pig jobs only)

Filtering Jobs

You filter jobs by selecting a time range and specifying a filter expression in the search box.

You can use the Time Range Selector or a duration link (

[30m](#) [1h](#) [2h](#) [6h](#) [12h](#) [1d](#) [7d](#) [30d](#)

) to set the time range.

Filter Expressions

Filter expressions specify which entries should display when you run the filter.

Filter Expressions

The simplest expression consists of three components:

- Attribute - Query language name of the attribute.
- Operator - Type of comparison between the attribute and the attribute value. Cloudera Manager supports the standard comparator operators =, !=, >, <, >=, <=, and RLIKE. (RLIKE performs regular expression matching as specified in the Java Pattern class documentation.) Numeric values can be compared with all operators. String values can be compared with =, !=, and RLIKE. Boolean values can be compared with = and !=.
- Value - The value of the attribute. The value depends on the type of the attribute. For a Boolean value, specify either true or false. When specifying a string value, enclose the value in double quotes.

You create compound filter expressions using the AND and OR operators. When more than one operator is used in an expression, AND is evaluated first, then OR. To change the order of evaluation, enclose subexpressions in parentheses.

Compound Expressions

To find all the jobs issued by the root user that ran for longer than ten seconds, use the expression:

```
user = "root" AND application_duration >= 100000.0
```

To find all the jobs that had more than 200 maps issued by users Jack or Jill, use the expression:

```
maps_completed >= 200.0 AND (user = "Jack" OR user = "Jill")
```

Choosing and Running a Filter

You can construct a filter, type a filter, or select a suggested or recently run filter.

Procedure

1. Do one of the following:

- Select a Suggested or Recently Run Filter: Click the  to the right of the Search button to display a list of sample and recently run filters, and select a filter. The filter text displays in the text box.
- Construct a Filter from the Workload Summary Attributes: Optionally, click Select Attributes to display a dialog box where you can chose attributes to display in the Workload Summary section. Select the checkbox

next to one or more attributes and click Close. Only attributes that support filtering appear in the Workload Summary section. See the Attributes table.

The attributes display in the Workload Summary section along with values or ranges of values that you can filter on. The values and ranges display as links with checkboxes. Select one or more checkboxes to add the range or value to the query. Click a link to run a query on that value or range. For example:

```
state = "SUCCEEDED" AND allocated_memory_seconds < 180000.0
```

Workload Summary

(For Completed Applications)

Allocated Memory Seconds 

- [120K - 180K](#) 1
- [240K - 300K](#) 1
- [360K - 420K](#) 1

Allocated VCore Seconds 

- [120 - 180](#) 1
- [240 - 300](#) 1
- [360 - 420](#) 1

Application State 

- [SUCCEEDED](#) 2
- [KILLED](#) 1

CPU Time 

- Type a Filter: Start typing or press Spacebar in the text box.
 - As you type, filter attributes matching the typed letter display. If you press Spacebar, standard filter attributes display. These suggestions are part of typeahead, which helps build valid queries. For information about the attribute name and supported values for each field, hover over the field in an existing query.
 - a. Select an attribute and press Enter.
 - b. Press Spacebar to display a drop-down list of operators.
 - c. Select an operator and press Enter.
 - d. Specify an attribute value. For attribute values that support typeahead, press the spacebar to display a drop-down list of values and press Enter. Alternatively, you can type a value.

- Click in the text box and press Enter or click Search. The list displays the results that match the specified filter. If the histograms are showing, they are redrawn to show only the values for the selected filter. The filter is added to the Recently Run list.

Filter Attributes

The table below describes filter attributes, their names as they are displayed in Cloudera Manager, their types, and descriptions.



Note: Only attributes where the Supports Filtering? column value is TRUE appear in the Workload Summary section.

Table 2: Attributes

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|--------|---------------------|---|
| Allocated Memory (allocated_mb) | NUMBER | FALSE | The sum of memory in MB allocated to the application's running containers. Called 'allocated_mb' in searches. |
| Allocated Memory Seconds (allocated_memory_seconds) | NUMBER | TRUE | The amount of memory the application has allocated (megabyte-seconds). Called 'allocated_memory_seconds' in searches. |
| Allocated VCores (allocated_vcores) | NUMBER | FALSE | The sum of virtual cores allocated to the application's running containers. Called 'allocated_vcores' in searches. |
| Allocated VCore Seconds (allocated_vcore_seconds) | NUMBER | TRUE | The amount of CPU resources the application has allocated (virtual core-seconds). Called 'allocated_vcore_seconds' in searches. |
| Application ID (application_id) | STRING | FALSE | The ID of the YARN application. Called 'application_id' in searches. |
| Application State (state) | STRING | TRUE | The state of this YARN application. This reflects the ResourceManager state while the application is running and the JobHistory Server state after the application has completed. Called 'state' in searches. |
| Application Tags (application_tags) | STRING | FALSE | A list of tags for the application. Called 'application_tags' in searches. |
| Application Type (application_type) | STRING | TRUE | The type of the YARN application. Called 'application_type' in searches. |
| Bytes Read (bytes_read) | BYTES | TRUE | Bytes read. Called 'bytes_read' in searches. |
| Bytes Written (bytes_written) | BYTES | TRUE | Bytes written. Called 'bytes_written' in searches. |
| Combine Input Records (combine_input_records) | NUMBER | TRUE | Combine input records. Called 'combine_input_records' in searches. |
| Combine Output Records (combine_output_records) | NUMBER | TRUE | Combine output records. Called 'combine_output_records' in searches. |
| Committed Heap (committed_heap_bytes) | BYTES | TRUE | Total committed heap usage. Called 'committed_heap_bytes' in searches. |
| Completed Maps and Reduces (tasks_completed) | NUMBER | TRUE | The number of completed map and reduce tasks in this MapReduce job. Called 'tasks_completed' in searches. Available only for running jobs. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|------------------------|--|
| CPU Allocation (vcores_millis) | NUMBER | TRUE | CPU allocation. This is the sum of 'vcores_millis_maps' and 'vcores_millis_reduces'. Called 'vcores_millis' in searches. |
| CPU Time (cpu_milliseconds) | MILLISECOND | TRUE | CPU time. Called 'cpu_milliseconds' in searches. |
| Data Local Maps (data_local_maps) | NUMBER | TRUE | Data local maps. Called 'data_local_maps' in searches. |
| Data Local Maps Percentage (data_local_maps_percentage) | NUMBER | TRUE | The number of data local maps as a percentage of the total number of maps. Called 'data_local_maps_percentage' in searches. |
| Diagnostics (diagnostics) | STRING | FALSE | Diagnostic information on the YARN application. If the diagnostic information is long, this may only contain the beginning of the information. Called 'diagnostics' in searches. |
| Duration (application_duration) | MILLISECOND | TRUE | How long YARN took to run this application. Called 'application_duration' in searches. |
| Executing (executing) | BOOLEAN | FALSE | Whether the YARN application is currently running. Called 'executing' in searches. |
| Failed Map and Reduce Attempts (failed_tasks_attempts) | NUMBER | TRUE | The number of failed map and reduce attempts for this MapReduce job. Called 'failed_tasks_attempts' in searches. Available only for failed jobs. |
| Failed Map Attempts (failed_map_attempts) | NUMBER | TRUE | The number of failed map attempts for this MapReduce job. Called 'failed_map_attempts' in searches. Available only for running jobs. |
| Failed Maps (num_failed_maps) | NUMBER | TRUE | Failed maps. Called 'num_failed_maps' in searches. |
| Failed Reduce Attempts (failed_reduce_attempts) | NUMBER | TRUE | The number of failed reduce attempts for this MapReduce job. Called 'failed_reduce_attempts' in searches. Available only for running jobs. |
| Failed Reduces (num_failed_reduces) | NUMBER | TRUE | Failed reduces. Called 'num_failed_reduces' in searches. |
| Failed Shuffles (failed_shuffle) | NUMBER | TRUE | Failed shuffles. Called 'failed_shuffle' in searches. |
| Failed Tasks (num_failed_tasks) | NUMBER | TRUE | The total number of failed tasks. This is the sum of 'num_failed_maps' and 'num_failed_reduces'. Called 'num_failed_tasks' in searches. |
| Fallow Map Slots Time (fallow_slots_millis_maps) | MILLISECOND | TRUE | Fallow map slots time. Called 'fallow_slots_millis_maps' in searches. |
| Fallow Reduce Slots Time (fallow_slots_millis_reduces) | MILLISECOND | TRUE | Fallow reduce slots time. Called 'fallow_slots_millis_reduces' in searches. |
| Fallow Slots Time (fallow_slots_millis) | MILLISECOND | TRUE | Total fallow slots time. This is the sum of 'fallow_slots_millis_maps' and 'fallow_slots_millis_reduces'. Called 'fallow_slots_millis' in searches. |
| File Bytes Read (file_bytes_read) | BYTES | TRUE | File bytes read. Called 'file_bytes_read' in searches. |
| File Bytes Written (file_bytes_written) | BYTES | TRUE | File bytes written. Called 'file_bytes_written' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|---|-------------|---------------------|---|
| File Large Read Operations (file_large_read_ops) | NUMBER | TRUE | File large read operations. Called 'file_large_read_ops' in searches. |
| File Read Operations (file_read_ops) | NUMBER | TRUE | File read operations. Called 'file_read_ops' in searches. |
| File Write Operations (file_write_ops) | NUMBER | TRUE | File write operations. Called 'file_large_write_ops' in searches. |
| Garbage Collection Time (gc_time_millis) | MILLISECOND | TRUE | Garbage collection time. Called 'gc_time_millis' in searches. |
| HDFS Bytes Read (hdfs_bytes_read) | BYTES | TRUE | HDFS bytes read. Called 'hdfs_bytes_read' in searches. |
| HDFS Bytes Written (hdfs_bytes_written) | BYTES | TRUE | HDFS bytes written. Called 'hdfs_bytes_written' in searches. |
| HDFS Large Read Operations (hdfs_large_read_ops) | NUMBER | TRUE | HDFS large read operations. Called 'hdfs_large_read_ops' in searches. |
| HDFS Read Operations (hdfs_read_ops) | NUMBER | TRUE | HDFS read operations. Called 'hdfs_read_ops' in searches. |
| HDFS Write Operations (hdfs_write_ops) | NUMBER | TRUE | HDFS write operations. Called 'hdfs_write_ops' in searches. |
| Hive Query ID (hive_query_id) | STRING | FALSE | If this MapReduce job ran as a part of a Hive query, this field contains the ID of the Hive query. Called 'hive_query_id' in searches. |
| Hive Query String (hive_query_string) | STRING | TRUE | If this MapReduce job ran as a part of a Hive query, this field contains the string of the query. Called 'hive_query_string' in searches. |
| Hive Sentry Subject Name (hive_sentry_subject_name) | STRING | TRUE | If this MapReduce job ran as a part of a Hive query on a secured cluster using impersonation, this field contains the name of the user that initiated the query. Called 'hive_sentry_subject_name' in searches. |
| Input Directory (input_dir) | STRING | TRUE | The input directory for this MapReduce job. Called 'input_dir' in searches. |
| Input Split Bytes (split_raw_bytes) | BYTES | TRUE | Input split bytes. Called 'split_raw_bytes' in searches. |
| Killed Map and Reduce Attempts (killed_tasks_attempts) | NUMBER | TRUE | The number of map and reduce attempts that were killed by user(s) for this MapReduce job. Called 'killed_tasks_attempts' in searches. Available only for killed jobs. |
| Killed Map Attempts (killed_map_attempts) | NUMBER | TRUE | The number of map attempts killed by user(s) for this MapReduce job. Called 'killed_map_attempts' in searches. Available only for running jobs. |
| Killed Reduce Attempts (killed_reduce_attempts) | NUMBER | TRUE | The number of reduce attempts killed by user(s) for this MapReduce job. Called 'killed_reduce_attempts' in searches. Available only for running jobs. |
| Launched Map Tasks (total_launched_maps) | NUMBER | TRUE | Launched map tasks. Called 'total_launched_maps' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|---------------------|---|
| Launched Reduce Tasks (total_launched_reduces) | NUMBER | TRUE | Launched reduce tasks. Called 'total_launched_reduces' in searches. |
| Map and Reduce Attempts in NEW State (new_tasks_attempts) | NUMBER | TRUE | The number of map and reduce attempts in NEW state for this MapReduce job. Called 'new_tasks_attempts' in searches. Available only for running jobs. |
| Map Attempts in NEW State (new_map_attempts) | NUMBER | TRUE | The number of map attempts in NEW state for this MapReduce job. Called 'new_map_attempts' in searches. Available only for running jobs. |
| Map Class (mapper_class) | STRING | TRUE | The class used by the map tasks in this MapReduce job. Called 'mapper_class' in searches. You can search for the mapper class using the class name alone, for example 'QuasiMonteCarlo\$QmcMapper', or the fully qualified classname, for example, 'org.apache.hadoop.examples.QuasiMonteCarlo\$QmcMapper'. |
| Map CPU Allocation (vcores_millis_maps) | NUMBER | TRUE | Map CPU allocation. Called 'vcores_millis_maps' in searches. |
| Map Input Records (map_input_records) | NUMBER | TRUE | Map input records. Called 'map_input_records' in searches. |
| Map Memory Allocation (mb_millis_maps) | NUMBER | TRUE | Map memory allocation. Called 'mb_millis_maps' in searches. |
| Map Output Bytes (map_output_bytes) | BYTES | TRUE | Map output bytes. Called 'map_output_bytes' in searches. |
| Map Output Materialized Bytes (map_output_materialized_bytes) | BYTES | TRUE | Map output materialized bytes. Called 'map_output_materialized_bytes' in searches. |
| Map Output Records (map_output_records) | NUMBER | TRUE | Map output records. Called 'map_output_records' in searches. |
| Map Progress (map_progress) | NUMBER | TRUE | The percentage of maps completed for this MapReduce job. Called 'map_progress' in searches. Available only for running jobs. |
| Maps Completed (maps_completed) | NUMBER | TRUE | The number of map tasks completed as a part of this MapReduce job. Called 'maps_completed' in searches. |
| Map Slots Time (slots_millis_maps) | MILLISECOND | TRUE | Total time spent by all maps in occupied slots. Called 'slots_millis_maps' in searches. |
| Maps Pending (maps_pending) | NUMBER | TRUE | The number of maps waiting to be run for this MapReduce job. Called 'maps_pending' in searches. Available only for running jobs. |
| Maps Running (maps_running) | NUMBER | TRUE | The number of maps currently running for this MapReduce job. Called 'maps_running' in searches. Available only for running jobs. |
| Maps Total (maps_total) | NUMBER | TRUE | The number of Map tasks in this MapReduce job. Called 'maps_total' in searches. |
| Memory Allocation (mb_millis) | NUMBER | TRUE | Total memory allocation. This is the sum of 'mb_millis_maps' and 'mb_millis_reduces'. Called 'mb_millis' in searches. |
| Merged Map Outputs (merged_map_outputs) | NUMBER | TRUE | Merged map outputs. Called 'merged_map_outputs' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|--------|---------------------|--|
| Name (name) | STRING | TRUE | Name of the YARN application. Called 'name' in searches. |
| Oozie Workflow ID (oozie_id) | STRING | FALSE | If this MapReduce job ran as a part of an Oozie workflow, this field contains the ID of the Oozie workflow. Called 'oozie_id' in searches. |
| Other Local Maps (other_local_maps) | NUMBER | TRUE | Other local maps. Called 'other_local_maps' in searches. |
| Other Local Maps Percentage (other_local_maps_percentage) | NUMBER | TRUE | The number of other local maps as a percentage of the total number of maps. Called 'other_local_maps_percentage' in searches. |
| Output Directory (output_dir) | STRING | TRUE | The output directory for this MapReduce job. Called 'output_dir' in searches. |
| Pending Maps and Reduces (tasks_pending) | NUMBER | TRUE | The number of maps and reduces waiting to be run for this MapReduce job. Called 'tasks_pending' in searches. Available only for running jobs. |
| Physical Memory (physical_memory_bytes) | BYTES | TRUE | Physical memory. Called 'physical_memory_bytes' in searches. |
| Pig Script ID (pig_id) | STRING | FALSE | If this MapReduce job ran as a part of a Pig script, this field contains the ID of the Pig script. Called 'pig_id' in searches. |
| Pool (pool) | STRING | TRUE | The name of the resource pool in which this application ran. Called 'pool' in searches. Within YARN, a pool is referred to as a queue. |
| Progress (progress) | NUMBER | TRUE | The progress reported by the application. Called 'progress' in searches. |
| Rack Local Maps (rack_local_maps) | NUMBER | TRUE | Rack local maps. Called 'rack_local_maps' in searches. |
| Rack Local Maps Percentage (rack_local_maps_percentage) | NUMBER | TRUE | The number of rack local maps as a percentage of the total number of maps. Called 'rack_local_maps_percentage' in searches. |
| Reduce Attempts in NEW State (new_reduce_attempts) | NUMBER | TRUE | The number of reduce attempts in NEW state for this MapReduce job. Called 'new_reduce_attempts' in searches. Available only for running jobs. |
| Reduce Class (reducer_class) | STRING | TRUE | The class used by the reduce tasks in this MapReduce job. Called 'reducer_class' in searches. You can search for the reducer class using the class name alone, for example 'QuasiMonteCarlo\$QmcReducer', or fully qualified classname, for example, 'org.apache.hadoop.examples.QuasiMonteCarlo\$QmcReducer'. |
| Reduce CPU Allocation (vcores_millis_reduces) | NUMBER | TRUE | Reduce CPU allocation. Called 'vcores_millis_reduces' in searches. |
| Reduce Input Groups (reduce_input_groups) | NUMBER | TRUE | Reduce input groups. Called 'reduce_input_groups' in searches. |
| Reduce Input Records (reduce_input_records) | NUMBER | TRUE | Reduce input records. Called 'reduce_input_records' in searches. |
| Reduce Memory Allocation (mb_millis_reduces) | NUMBER | TRUE | Reduce memory allocation. Called 'mb_millis_reduces' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|---------------------|---|
| Reduce Output Records (reduce_output_records) | NUMBER | TRUE | Reduce output records. Called 'reduce_output_records' in searches. |
| Reduce Progress (reduce_progress) | NUMBER | TRUE | The percentage of reduces completed for this MapReduce job. Called 'reduce_progress' in searches. Available only for running jobs. |
| Reduces Completed (reduces_completed) | NUMBER | TRUE | The number of reduce tasks completed as a part of this MapReduce job. Called 'reduces_completed' in searches. |
| Reduce Shuffle Bytes (reduce_shuffle_bytes) | BYTES | TRUE | Reduce shuffle bytes. Called 'reduce_shuffle_bytes' in searches. |
| Reduce Slots Time (slots_millis_reduces) | MILLISECOND | TRUE | Total time spent by all reduces in occupied slots. Called 'slots_millis_reduces' in searches. |
| Reduces Pending (reduces_pending) | NUMBER | TRUE | The number of reduces waiting to be run for this MapReduce job. Called 'reduces_pending' in searches. Available only for running jobs. |
| Reduces Running (reduces_running) | NUMBER | TRUE | The number of reduces currently running for this MapReduce job. Called 'reduces_running' in searches. Available only for running jobs. |
| Reduces Total (reduces_total) | NUMBER | TRUE | The number of reduce tasks in this MapReduce job. Called 'reduces_total' in searches. |
| Running Containers (running_containers) | NUMBER | FALSE | The number of containers currently running for the application. Called 'running_containers' in searches. |
| Running Map and Reduce Attempts (running_tasks_attempts) | NUMBER | TRUE | The number of map and reduce attempts currently running for this MapReduce job. Called 'running_tasks_attempts' in searches. Available only for running jobs. |
| Running Map Attempts (running_map_attempts) | NUMBER | TRUE | The number of running map attempts for this MapReduce job. Called 'running_map_attempts' in searches. Available only for running jobs. |
| Running MapReduce Application Information Retrieval Duration. (running_application_info_retrieval_time) | NUMBER | TRUE | How long it took, in seconds, to retrieve information about the MapReduce application. |
| Running Maps and Reduces (tasks_running) | NUMBER | TRUE | The number of maps and reduces currently running for this MapReduce job. Called 'tasks_running' in searches. Available only for running jobs. |
| Running Reduce Attempts (running_reduce_attempts) | NUMBER | TRUE | The number of running reduce attempts for this MapReduce job. Called 'running_reduce_attempts' in searches. Available only for running jobs. |
| Service Name (service_name) | STRING | FALSE | The name of the YARN service. Called 'service_name' in searches. |
| Shuffle Bad ID Errors (shuffle_errors_bad_id) | NUMBER | TRUE | Shuffle bad ID errors. Called 'shuffle_errors_bad_id' in searches. |
| Shuffle Connection Errors (shuffle_errors_connection) | NUMBER | TRUE | Shuffle connection errors. Called 'shuffle_errors_connection' in searches. |
| Shuffled Maps (shuffled_maps) | NUMBER | TRUE | Shuffled maps. Called 'shuffled_maps' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|---|-------------|---------------------|--|
| Shuffle IO Errors (shuffle_errors_io) | NUMBER | TRUE | Shuffle IO errors. Called 'shuffle_errors_io' in searches. |
| Shuffle Wrong Length Errors (shuffle_errors_wrong_length) | NUMBER | TRUE | Shuffle wrong length errors. Called 'shuffle_errors_wrong_length' in searches. |
| Shuffle Wrong Map Errors (shuffle_errors_wrong_map) | NUMBER | TRUE | Shuffle wrong map errors. Called 'shuffle_errors_wrong_map' in searches. |
| Shuffle Wrong Reduce Errors (shuffle_errors_wrong_reduce) | NUMBER | TRUE | Shuffle wrong reduce errors. Called 'shuffle_errors_wrong_reduce' in searches. |
| Slots Time (slots_millis) | MILLISECOND | TRUE | Total slots time. This is the sum of 'slots_millis_maps' and 'slots_millis_reduces'. Called 'slots_millis' in searches. |
| Spilled Records (spilled_records) | NUMBER | TRUE | Spilled Records. Called 'spilled_records' in searches. |
| Successful Map and Reduce Attempts (successful_tasks_attempts) | NUMBER | TRUE | The number of successful map and reduce attempts for this MapReduce job. Called 'successful_tasks_attempts' in searches. Available only for successful jobs. |
| Successful Map Attempts (successful_map_attempts) | NUMBER | TRUE | The number of successful map attempts for this MapReduce job. Called 'successful_map_attempts' in searches. Available only for running jobs. |
| Successful Reduce Attempts (successful_reduce_attempts) | NUMBER | TRUE | The number of successful reduce attempts for this MapReduce job. Called 'successful_reduce_attempts' in searches. Available only for running jobs. |
| Total Maps and Reduces Number (total_task_num) | NUMBER | TRUE | The number of map and reduce tasks in this MapReduce job. Called 'tasks_total' in searches. Available only for running jobs. |
| Total Tasks (total_launched_tasks) | NUMBER | TRUE | The total number of tasks. This is the sum of 'total_launched_maps' and 'total_launched_reduces'. Called 'total_launched_tasks' in searches. |
| Tracking Url (tracking_url) | STRING | FALSE | The MapReduce application tracking URL. |
| Uberized Job (uberized) | BOOLEAN | FALSE | Whether this MapReduce job is uberized - running completely in the ApplicationMaster. Called 'uberized' in searches. Available only for running jobs. |
| Unused Memory Seconds (unused_memory_seconds) | NUMBER | TRUE | The amount of memory the application has allocated but not used (megabyte-seconds). This metric is calculated hourly if container usage metric aggregation is enabled. Called 'unused_memory_seconds' in searches. |
| Unused VCore Seconds (unused_vcore_seconds) | NUMBER | TRUE | The amount of CPU resources the application has allocated but not used (virtual core-seconds). This metric is calculated hourly if container usage metric aggregation is enabled. Called 'unused_vcore_seconds' in searches. |
| Used Memory Max (used_memory_max) | NUMBER | TRUE | The maximum container memory usage for a YARN application. This metric is calculated hourly if container usage metric aggregation is enabled and a Cloudera Manager Container Usage Metrics Directory is specified. |
| User (user) | STRING | TRUE | The user who ran the YARN application. Called 'user' in searches. |
| Virtual Memory (virtual_memory_bytes) | BYTES | TRUE | Virtual memory. Called 'virtual_memory_bytes' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|---------------------|---|
| Work CPU Time (cm_cpu_milliseconds) | MILLISECOND | TRUE | Attribute measuring the sum of CPU time used by all threads of the query, in milliseconds. Called 'work_cpu_time' in searches. For Impala queries, CPU time is calculated based on the 'TotalCpuTime' metric. For YARN MapReduce applications, this is calculated from the 'cpu_milliseconds' metric. |

Examples

Consider the following filter expressions: `user = "root"`, `rowsProduced > 0`, `fileFormats RLIKE ".TEXT.*"`, and `executing = true`. In the examples:

- The filter attributes are `user`, `rowsProduced`, `fileFormats`, and `executing`.
- The operators are `=`, `>`, and `RLIKE`.
- The filter values are `root`, `0`, `.TEXT.*`, and `true`.

Sending Diagnostic Data to Cloudera for YARN Applications

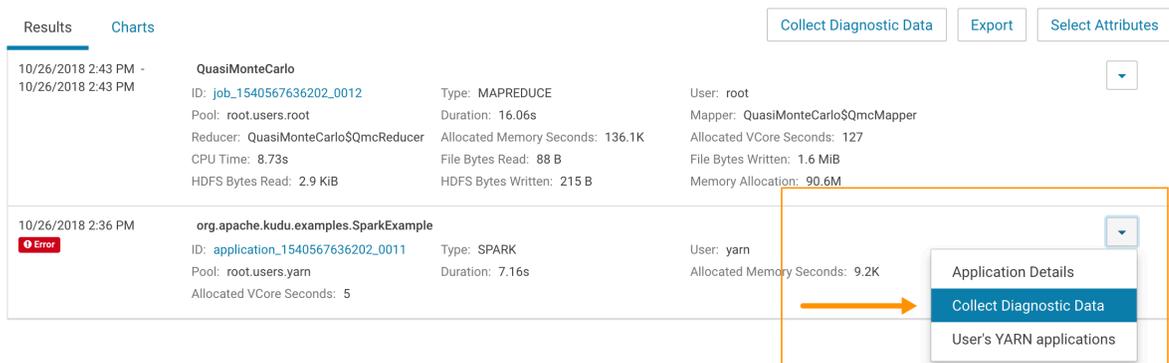
You can send diagnostic data collected from YARN applications, including metadata, configurations, and log data, to Cloudera Support for analysis. Include a support ticket number if one exists to enable Cloudera Support to address the issue more quickly and efficiently.

About this task

Minimum Required Role: [Cluster Administrator](#) (also provided by Full Administrator) This feature is not available when using Cloudera Manager to manage Data Hub clusters.

Procedure

1. From the YARN page in Cloudera Manager, click the Applications menu.
2. Collect diagnostics data. There are two ways to do this:
 - To collect data from all applications that are visible in the list, click the top Collect Diagnostics Data button on the upper right, above the list of YARN applications.
 - To collect data from only one specific application, click the down arrow on the right-hand end of the row of the application and select Collect Diagnostics Data.



3. In the Send YARN Applications Diagnostic Data dialog box, provide the following information:
 - If applicable, the Cloudera Support ticket number of the issue being experienced on the cluster.
 - Optionally, add a comment to help the support team understand the issue.
4. Click the checkbox Send Diagnostic Data to Cloudera.

5. Click the button Collect and Send Diagnostic Data.



Note: Passwords from configuration will not be retrieved.

Monitoring Impala Queries

The Impala Queries page displays information about Impala queries that are running and have run in your cluster. You can filter the queries by time period and by specifying simple filtering expressions.



Note: The Impala query monitoring feature requires Impala 1.0.1 and higher.

Viewing Queries

You can view queries, filter queries, and more from the Impala service **Queries** tab.

Do one of the following:

- Select Clusters *Cluster name* *Impala service name* **Queries** .
- On the **Home Status** tab, select *Impala service name* and click the **Queries** tab.

The Impala queries run during the selected time range display in the **Results** tab.

You can also perform the following actions on this page:

Table 3: Viewing Queries Actions

| Action | Description |
|--|--|
| Filter the displayed queries | Create filter expressions manually, select preconfigured filters, or use the Workload Summary section to build a query interactively. |
| Select additional attributes for display. | Click Select Attributes. Selected attributes also display as available filters in the Workload Summary section. To display information about attributes, hover over a field label. . Only attributes that support filtering appear in the Workload Summary section. |
| View a histogram of the attribute values. | Click the  icon to the right of each attribute displayed in the Workload Summary section. |
| Display charts based on the filter expression and selected attributes. | Click the Charts tab. |
| View charts that help identify whether Impala best practices are being followed. | Click the Best Practices link. |
| Export a JSON file with the query results that you can use for further analysis. | Click Export. |

Configuring Impala Query Monitoring

You can configure the visibility of the Impala query results and the size of the storage allocated to Impala query results.

About this task

To configure whether admin or non-admin users can view all queries, only that user's queries, or no queries:

Minimum Required Role: **Configurator** (also provided by Cluster Administrator, Limited Cluster Administrator , and Full Administrator)

Procedure

1. Go to the Impala service.
2. Click the Configuration tab.
3. Select Scope *Impala service_name* (Service-Wide).
4. Click the Monitoring category.
5. Set the Visibility Settings properties for admin and non-admin users.
6. Enter a Reason for change, and then click Save Changes to commit the changes.
7. Click the Cloudera Manager logo to return to the Home page.
8. Click the  icon that is next to any stale services to invoke the cluster restart wizard.

Impala Best Practices

The **Impala Best Practices** page contains charts that include description of each best practice and how to determine if it is being followed.

To open the Impala Best Practices page, click the Best Practices tab on the Impala service page. See the Impala documentation for more detail on each best practice and for additional best practices.

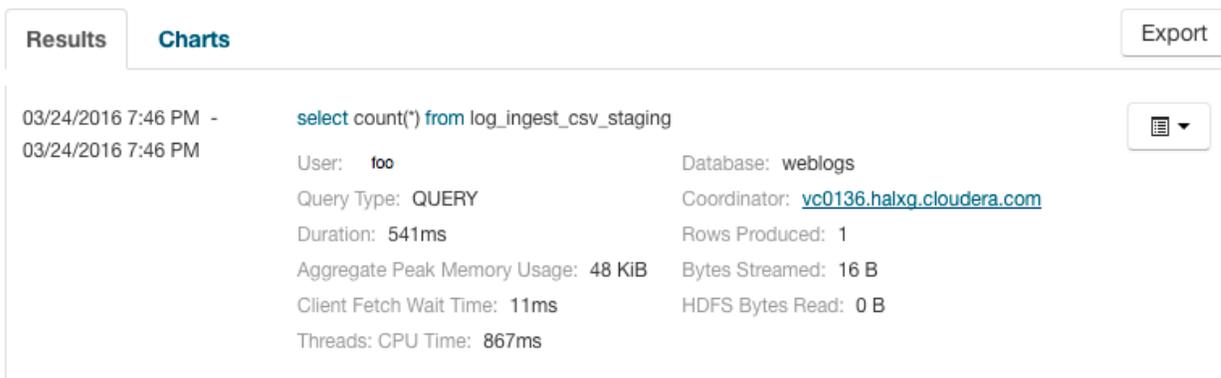
Adjust the time range to see data on queries run at different times. Click the charts to get more detail on individual queries. Use the filter box at the top right of the Best Practices page to adjust which data is shown on the page. For example, to see just the queries that took more than ten seconds, make the filter `query_duration > 10s`.

Create a trigger based on any best practice by choosing Create Trigger from the individual chart drop-down menu.

Results Tab

Queries appear on the Results tab, with the most recent at the top. Each query has summary and detail information.

A query summary includes the following default attributes: start and end timestamps, statement, duration, rows produced, user, coordinator, database, and query type. For example:



The screenshot shows the 'Results' tab in a Cloudera interface. It displays a query summary for a query executed on 03/24/2016 at 7:46 PM. The query statement is `select count(*) from log_ingest_csv_staging`. The summary includes the following attributes:

- User: `foo`
- Database: `weblogs`
- Query Type: `QUERY`
- Coordinator: `vc0136.halxg.cloudera.com`
- Duration: `541ms`
- Rows Produced: `1`
- Aggregate Peak Memory Usage: `48 KiB`
- Bytes Streamed: `16 B`
- Client Fetch Wait Time: `11ms`
- HDFS Bytes Read: `0 B`
- Threads: `CPU Time: 867ms`

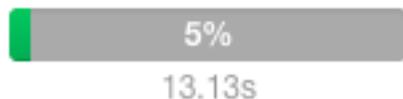
You can add additional attributes to the summary by clicking the Attribute Selector. In each query summary, the query statement is truncated if it is too long to display. To display the entire statement, click . The query entry expands to display the entire query string. To collapse the query display, click . To display information about query attributes and possible values, hover over a field in a query. For example:

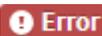
The type of this query. Possible values are QUERY, DDL and DML. Called "queryType" in searches.

Query Type: **QUERY**

A running job displays a progress bar under the starting timestamp:

03/25/2016 10:03 AM



If an error occurred while processing the query,  displays under the complete timestamp.

Use the Actions drop-down menu  to the right of each query listing to do the following. (Not all options display, depending on the type of job.)

- Query Details – Opens a details page for the job.
- User's Impala Queries – Displays a list of queries run by the user for the current job.
- Cancel (running queries only) – Cancel a running query (administrators only). Canceling a running query creates an audit event. When you cancel a query,  replaces the progress bar.
- Queries in the same YARN pool – Displays queries that use the same resource pool.

Filtering Queries

You filter queries by selecting a time range and specifying a filter expression in the search box.

You filter queries by selecting a time range and specifying a filter expression in the search box.

You can use the Time Range Selector or a duration link (

[30m](#) [1h](#) [2h](#) [6h](#) [12h](#) [1d](#) [7d](#) [30d](#)) to set the time range.

Filter Expressions

Filter expressions specify which entries should display when you run the filter.

The simplest expression consists of three components:

- Attribute - Query language name of the attribute.
- Operator - Type of comparison between the attribute and the attribute value. Cloudera Manager supports the standard comparator operators =, !=, >, <, >=, <=, and RLIKE. (RLIKE performs regular expression matching as specified in the Java Pattern class documentation.) Numeric values can be compared with all operators. String values can be compared with =, !=, and RLIKE. Boolean values can be compared with = and !=.
- Value - The value of the attribute. The value depends on the type of the attribute. For a Boolean value, specify either true or false. When specifying a string value, enclose the value in double quotes.

You create compound filter expressions using the AND and OR operators. When more than one operator is used in an expression, AND is evaluated first, then OR. To change the order of evaluation, enclose subexpressions in parentheses.

Compound Expressions

To find all the queries issued by the root user that produced over 100 rows, use the expression:

```
user = "root" AND rowsProduced > 100
```

To find all the executing queries issued by users Jack or Jill, use the expression:

```
executing = true AND (user = "Jack" OR user = "Jill")
```

Filter Attributes

The following table includes available filter attributes and their names in Cloudera Manager, types, and descriptions.



Note: Only attributes for which the Supports Filtering? column value is TRUE appear in the Workload Summary section.

Table 4: Attributes

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|------------------|---------------------|---|
| Admission Result (admission_result) | STRING | TRUE | The result of admission, whether immediately, queued, rejected, or timed out. Called 'admission_result' in searches. |
| Admission Wait Time (admission_wait) | MILLISECOND | TRUE | The time from submission for admission to completion of admission. Called 'admission_wait' in searches. |
| Aggregate Peak Memory Usage (memory_aggregate_peak) | BYTES | TRUE | The highest amount of memory allocated by this query at a particular time across all nodes. Called 'memory_aggregate_peak' in searches. |
| Bytes Streamed (bytes_streamed) | BYTES | TRUE | The total number of bytes sent between Impala Daemons while processing this query. Called 'bytes_streamed' in searches. |
| Client Fetch Wait Time (client_fetch_wait_time) | MILLISECOND | TRUE | The total amount of time the query spent waiting for the client to fetch row data. Called 'client_fetch_wait_time' in searches. |
| Client Fetch Wait Time Percentage (client_fetch_wait_time_percentage) | NUMBER | TRUE | The total amount of time the query spent waiting for the client to fetch row data divided by the query duration. Called 'client_fetch_wait_time_percentage' in searches. |
| Connected User (connected_user) | STRING | TRUE | The user who created the Impala session that issued this query. This is distinct from 'user' only if delegation is in use. Called 'connected_user' in searches. |
| Coordinator (coordinator_host_id) | STRING | TRUE | The host coordinating this query. Called 'coordinator_host_id' in searches. |
| Database (database) | STRING | TRUE | The database on which the query was run. Called 'database' in searches. |
| DDL Type (ddl_type) | STRING | TRUE | The type of DDL query. Called 'ddl_type' in searches. |
| Delegated User (delegated_user) | STRING | TRUE | The effective user for the query. This is set only if delegation is in use. Called 'delegated_user' in searches. |
| Duration (query_duration) | MILLISECOND | TRUE | The duration of the query in milliseconds. Called 'query_duration' in searches. |
| Estimated per Node Peak Memory (estimated_per_node_peak_memory) | BYTES | TRUE | The planning process's estimate of per-node peak memory usage for the query. Called 'estimated_per_node_peak_memory' in searches. |
| Executing (executing) | BOOLEAN | FALSE | Whether the query is currently executing. Called 'executing' in searches. |
| File Formats (file_formats) | STRING | FALSE | An alphabetically sorted list of all the file formats used in the query. Called 'file_formats' in searches. |
| HBase Bytes Read (hbase_bytes_read) | BYTES | TRUE | The total number of bytes read from HBase by this query. Called 'hbase_bytes_read' in searches. |
| HBase Scanner Average Read Throughput (hbase_scanner_average_bytes_read_per_second) | BYTES_PER_SECOND | TRUE | The average HBase scanner read throughput for this query. This is computed by dividing the total bytes read from HBase by the total time spent reading by all HBase scanners. Called 'hbase_scanner_average_bytes_read_per_second' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|------------------|---------------------|--|
| HDFS Average Scan Range (hdfs_average_scan_range) | BYTES | TRUE | The average HDFS scan range size for this query. HDFS scan nodes that contained only a single scan range are not included in this computation. Low numbers for a query might indicate reading many small files which negatively impacts performance. Called 'hdfs_average_scan_range' in searches. |
| HDFS Bytes Read (hdfs_bytes_read) | BYTES | TRUE | The total number of bytes read from HDFS by this query. Called 'hdfs_bytes_read' in searches. |
| HDFS Bytes Read From Cache (hdfs_bytes_read_from_cache) | BYTES | TRUE | The total number of bytes read from HDFS that were read from the HDFS cache. This is only for completed queries. Called 'hdfs_bytes_read_from_cache' in searches. |
| HDFS Bytes Read From Cache Percentage (hdfs_bytes_read_from_cache_percentage) | NUMBER | TRUE | The percentage of all bytes read by this query that were read from the HDFS cache. This is only for completed queries. Called 'hdfs_bytes_read_from_cache_percentage' in searches. |
| HDFS Bytes Skipped (hdfs_bytes_skipped) | BYTES | TRUE | The total number of bytes that had to be skipped by this query while reading from HDFS. Any number above zero may indicate a problem. Called 'hdfs_bytes_skipped' in searches. |
| HDFS Bytes Written (hdfs_bytes_written) | BYTES | TRUE | The total number of bytes written to HDFS by this query. Called 'hdfs_bytes_written' in searches. |
| HDFS Local Bytes Read (hdfs_bytes_read_local) | BYTES | TRUE | The total number of local bytes read from HDFS by this query. This is only for completed queries. Called 'hdfs_bytes_read_local' in searches. |
| HDFS Local Bytes Read Percentage (hdfs_bytes_read_local_percentage) | NUMBER | TRUE | The percentage of all bytes read from HDFS by this query that were local. This is only for completed queries. Called 'hdfs_bytes_read_local_percentage' in searches. |
| HDFS Remote Bytes Read (hdfs_bytes_read_remote) | BYTES | TRUE | The total number of remote bytes read from HDFS by this query. This is only for completed queries. Called 'hdfs_bytes_read_remote' in searches. |
| HDFS Remote Bytes Read Percentage (hdfs_bytes_read_remote_percentage) | NUMBER | TRUE | The percentage of all bytes read from HDFS by this query that were remote. This is only for completed queries. Called 'hdfs_bytes_read_remote_percentage' in searches. |
| HDFS Scanner Average Read Throughput (hdfs_scanner_average_bytes_read_per_second) | BYTES_PER_SECOND | TRUE | The average HDFS scanner read throughput for this query. This is computed by dividing the total bytes read from HDFS by the total time spent reading by all HDFS scanners. Called 'hdfs_scanner_average_bytes_read_per_second' in searches. |
| HDFS Short Circuit Bytes Read (hdfs_bytes_read_short_circuit) | BYTES | TRUE | The total number of bytes read from HDFS by this query that used short-circuit reads. This is only for completed queries. Called 'hdfs_bytes_read_short_circuit' in searches. |
| HDFS Short Circuit Bytes Read Percentage (hdfs_bytes_read_short_circuit_percentage) | NUMBER | TRUE | The percentage of all bytes read from HDFS by this query that used short-circuit reads. This is only for completed queries. Called 'hdfs_bytes_read_short_circuit_percentage' in searches. |
| Impala Version (impala_version) | STRING | TRUE | The version of the Impala Daemon coordinating this query. Called 'impala_version' in searches. |
| Memory Accrual (memory_accrual) | BYTE_SECONDS | TRUE | The total accrued memory usage by the query. This is computed by multiplying the average aggregate memory usage of the query by the query's duration. Called 'memory_accrual' in searches. |
| Memory Spilled (memory_spilled) | BYTES | TRUE | Amount of memory spilled to disk. Called 'memory_spilled' in searches. |
| Network Address (network_address) | STRING | TRUE | The network address that issued this query. Called 'network_address' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|------------------------|--|
| Node with Peak Memory Usage (memory_per_node_peak_node) | STRING | TRUE | The node with the highest peak memory usage for this query. See Per Node Peak Memory Usage for the actual peak value. Called 'memory_per_node_peak_node' in searches. |
| Out of Memory (oom) | BOOLEAN | TRUE | Whether the query ran out of memory. Called 'oom' in searches. |
| Per Node Peak Memory Usage (memory_per_node_peak) | BYTES | TRUE | The highest amount of memory allocated by any single node that participated in this query. See Node with Peak Memory Usage for the name of the peak node. Called 'memory_per_node_peak' in searches. |
| Planning Wait Time (planning_wait_time) | MILLISECOND | TRUE | The total amount of time the query spent waiting for planning to complete. Called 'planning_wait_time' in searches. |
| Planning Wait Time Percentage (planning_wait_time_percentage) | NUMBER | TRUE | The total amount of time the query spent waiting for planning to complete divided by the query duration. Called 'planning_wait_time_percentage' in searches. |
| Pool (pool) | STRING | TRUE | The name of the resource pool in which this query executed. Called 'pool' in searches. If YARN is in use, this corresponds to a YARN pool. Within YARN, a pool is referred to as a queue. |
| Query ID (query_id) | STRING | FALSE | The id of this query. Called 'query_id' in searches. |
| Query State (query_state) | STRING | TRUE | The current state of the query (running, finished, and so on). Called 'query_state' in searches. |
| Query Status (query_status) | STRING | TRUE | The status of the query. If the query hasn't failed the status will be 'OK', otherwise it will provide more information on the cause of the failure. Called 'query_status' in searches. |
| Query Type (query_type) | STRING | TRUE | The type of the query's SQL statement (DML, DDL, Query). Called 'query_type' in searches. |
| Resource Reservation Wait Time (resources_reserved_wait_time) | MILLISECOND | TRUE | The total amount of time the query spent waiting for pool resources to become available. Called 'resources_reserved_wait_time' in searches. |
| Resource Reservation Wait Time Percentage (resources_reserved_wait_time_percentage) | NUMBER | TRUE | The total amount of time the query spent waiting for pool resources to become available divided by the query duration. Called 'resources_reserved_wait_time_percentage' in searches. |
| Rows Inserted (rows_inserted) | NUMBER | TRUE | The number of rows inserted by the query. Called 'rows_inserted' in searches. |
| Rows Produced (rows_produced) | NUMBER | TRUE | The number of rows produced by the query. Called 'rows_produced' in searches. |
| Service Name (service_name) | STRING | FALSE | The name of the Impala service. Called 'service_name' in searches. |
| Session ID (session_id) | STRING | TRUE | The ID of the session that issued this query. Called 'session_id' in searches. |
| Session Type (session_type) | STRING | TRUE | The type of the session that issued this query. Called 'session_type' in searches. |
| Statement (statement) | STRING | FALSE | The query's SQL statement. Called 'statement' in searches. |

| Display Name (Attribute Name) | Type | Supports Filtering? | Description |
|--|-------------|---------------------|--|
| Statistics Missing (stats_missing) | BOOLEAN | TRUE | Whether the query was flagged with missing table or column statistics warning during the planning process. Called 'stats_missing' in searches. |
| Threads: CPU Time (thread_cpu_time) | MILLISECOND | TRUE | The sum of the CPU time used by all threads of the query. Called 'thread_cpu_time' in searches. |
| Threads: CPU Time Percentage (thread_cpu_time_percentage) | NUMBER | TRUE | The sum of the CPU time used by all threads of the query divided by the total thread time. Called 'thread_cpu_time_percentage' in searches. |
| Threads: Network Receive Wait Time (thread_network_receive_wait_time) | MILLISECOND | TRUE | The sum of the time spent waiting to receive data over the network by all threads of the query. A query will almost always have some threads waiting to receive data from other nodes in the query's execution tree. Unlike other wait times, network receive wait time does not usually indicate an opportunity for improving a query's performance. Called 'thread_network_receive_wait_time' in searches. |
| Threads: Network Receive Wait Time Percentage (thread_network_receive_wait_time_percentage) | NUMBER | TRUE | The sum of the time spent waiting to receive data over the network by all threads of the query divided by the total thread time. A query will almost always have some threads waiting to receive data from other nodes in the query's execution tree. Unlike other wait times, network receive wait time does not usually indicate an opportunity for improving a query's performance. Called 'thread_network_receive_wait_time_percentage' in searches. |
| Threads: Network Send Wait Time (thread_network_send_wait_time) | MILLISECOND | TRUE | The sum of the time spent waiting to send data over the network by all threads of the query. Called 'thread_network_send_wait_time' in searches. |
| Threads: Network Send Wait Time Percentage (thread_network_send_wait_time_percentage) | NUMBER | TRUE | The sum of the time spent waiting to send data over the network by all threads of the query divided by the total thread time. Called 'thread_network_send_wait_time_percentage' in searches. |
| Threads: Storage Wait Time (thread_storage_wait_time) | MILLISECOND | TRUE | The sum of the time spent waiting for storage by all threads of the query. Called 'thread_storage_wait_time' in searches. |
| Threads: Storage Wait Time Percentage (thread_storage_wait_time_percentage) | NUMBER | TRUE | The sum of the time spent waiting for storage by all threads of the query divided by the total thread time. Called 'thread_storage_wait_time_percentage' in searches. |
| Threads: Total Time (thread_total_time) | MILLISECOND | TRUE | The sum of thread CPU, storage wait and network wait times used by all threads of the query. Called 'thread_total_time' in searches. |
| User (user) | STRING | TRUE | The effective user for the query. This is the delegated user if delegation is in use. Otherwise, this is the connected user. Called 'user' in searches. |
| Work CPU Time (cm_cpu_milliseconds) | MILLISECOND | TRUE | Attribute measuring the sum of CPU time used by all threads of the query, in milliseconds. Called 'work_cpu_time' in searches. For Impala queries, CPU time is calculated based on the 'TotalCpuTime' metric. For YARN MapReduce applications, this is calculated from the 'cpu_milliseconds' metric. |

Examples

Consider the following filter expressions: `user = "root"`, `rowsProduced > 0`, `fileFormats RLIKE ".TEXT.*"`, and `executing = true`. In the examples:

- The filter attributes are `user`, `rowsProduced`, `fileFormats`, and `executing`.
- The operators are `=`, `>`, and `RLIKE`.
- The filter values are `root`, `0`, `.TEXT.*`, and `true`.

Choosing and Running a Filter

You can construct a filter, type a filter, or select a suggested or recently run filter.

Procedure

1. Do one of the following:

- Select a Suggested or Recently Run Filter: Click the  to the right of the Search button to display a list of sample and recently run filters, and select a filter. The filter text displays in the text box.
- Construct a Filter from the Workload Summary Attributes: Optionally, click Select Attributes to display a dialog box where you can chose which attributes to display in the Workload Summary section. Select the checkbox next to one or more attributes, and click Close.

The attributes display in the Workload Summary section along with values or ranges of values that you can filter on. The values and ranges display as links with checkboxes. Select one or more checkboxes to add the range or value to the query. Click a link to run a query on that value or range. For example:

```
bytes_streamed < 60.0 AND memory_aggregate_peak < 100000.0
```

Workload Summary

(For Completed Queries)

Aggregate Peak Memory Usage

| | | |
|-------------------------------------|--------------------------------------|-----|
| <input checked="" type="checkbox"/> | 12 KiB - 97.7 KiB | 18 |
| <input type="checkbox"/> | 97.7 KiB - 976.6 KiB | 8 |
| <input type="checkbox"/> | 976.6 KiB - 9.5 MiB | 55 |
| <input type="checkbox"/> | 9.5 MiB - 95.4 MiB | 222 |
| <input type="checkbox"/> | 95.4 MiB - 953.7 MiB | 62 |
| <input type="checkbox"/> | 953.7 MiB - 9.3 GiB | 42 |
| <input type="checkbox"/> | 9.3 GiB - 34.1 GiB | 9 |

Bytes Streamed

| | | |
|-------------------------------------|--------------------------------------|-----|
| <input checked="" type="checkbox"/> | 0 B - 60 B | 36 |
| <input type="checkbox"/> | 60 B - 600 B | 173 |
| <input type="checkbox"/> | 600 B - 5.9 KiB | 49 |
| <input type="checkbox"/> | 5.9 KiB - 58.6 KiB | 66 |
| <input type="checkbox"/> | 58.6 KiB - 585.9 KiB | 96 |
| <input type="checkbox"/> | 585.9 KiB - 5.7 MiB | 19 |
| <input type="checkbox"/> | 5.7 MiB - 5.1 GiB | 9 |

- Type a Filter: Start typing or press Spacebar in the text box.

As you type, filter attributes matching the typed letter display. If you press Spacebar, standard filter attributes display. These suggestions are part of typeahead, which helps build valid queries. For information about the attribute name and supported values for each field, hover over the field in an existing query.

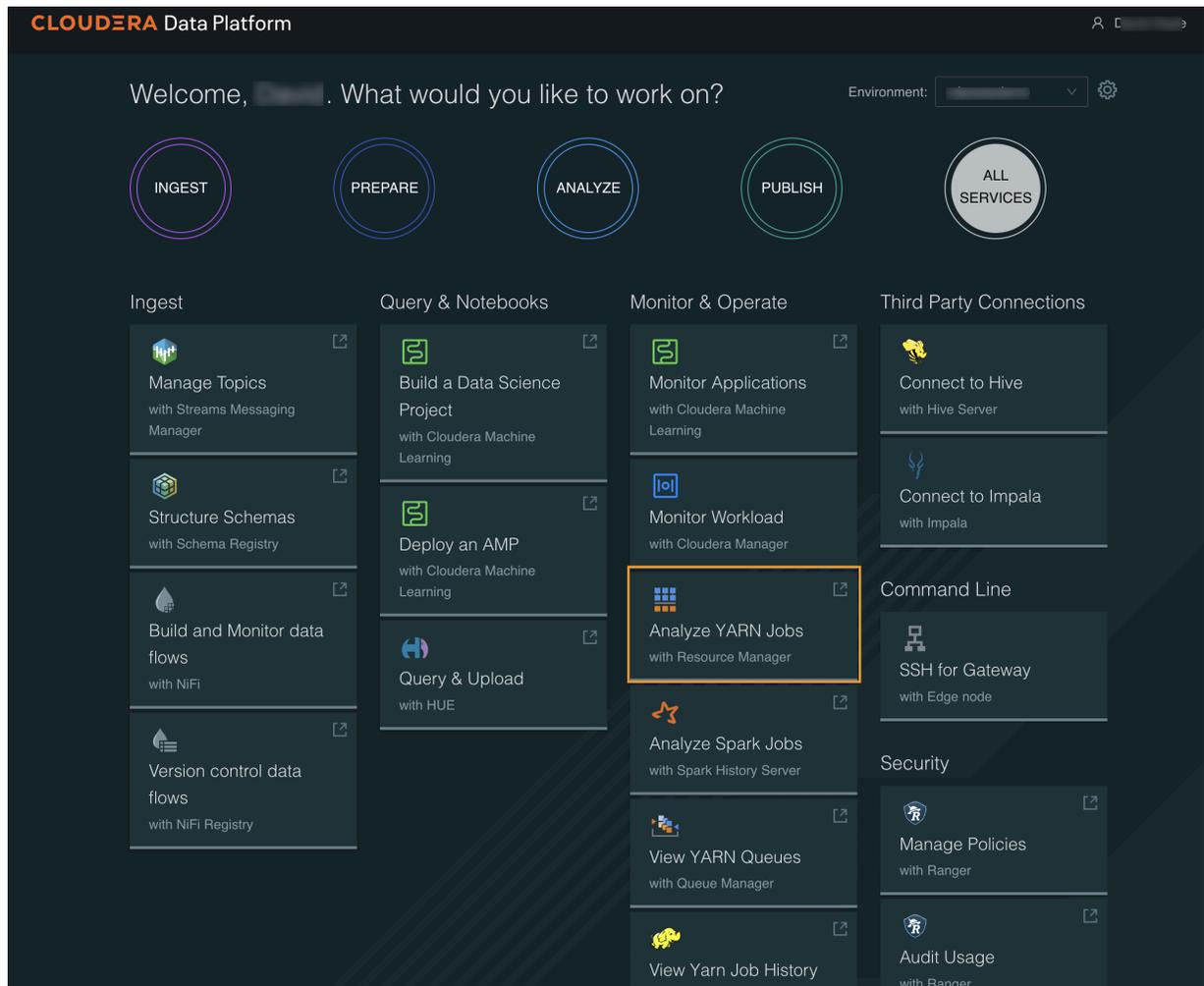
- a. Select an attribute and press Enter.
 - b. Press Spacebar to display a drop-down list of operators.
 - c. Select an operator and press Enter.
 - d. Specify an attribute value. For attribute values that support typeahead, press the spacebar to display a drop-down list of values and press Enter. Alternatively, you can type a value.
2. Click in the text box and press Enter or click Search. The list displays the results that match the specified filter. The Workload Summary section refreshes to show only the values for the selected filter. The filter is added to the Recently Run list.

Analyzing YARN jobs

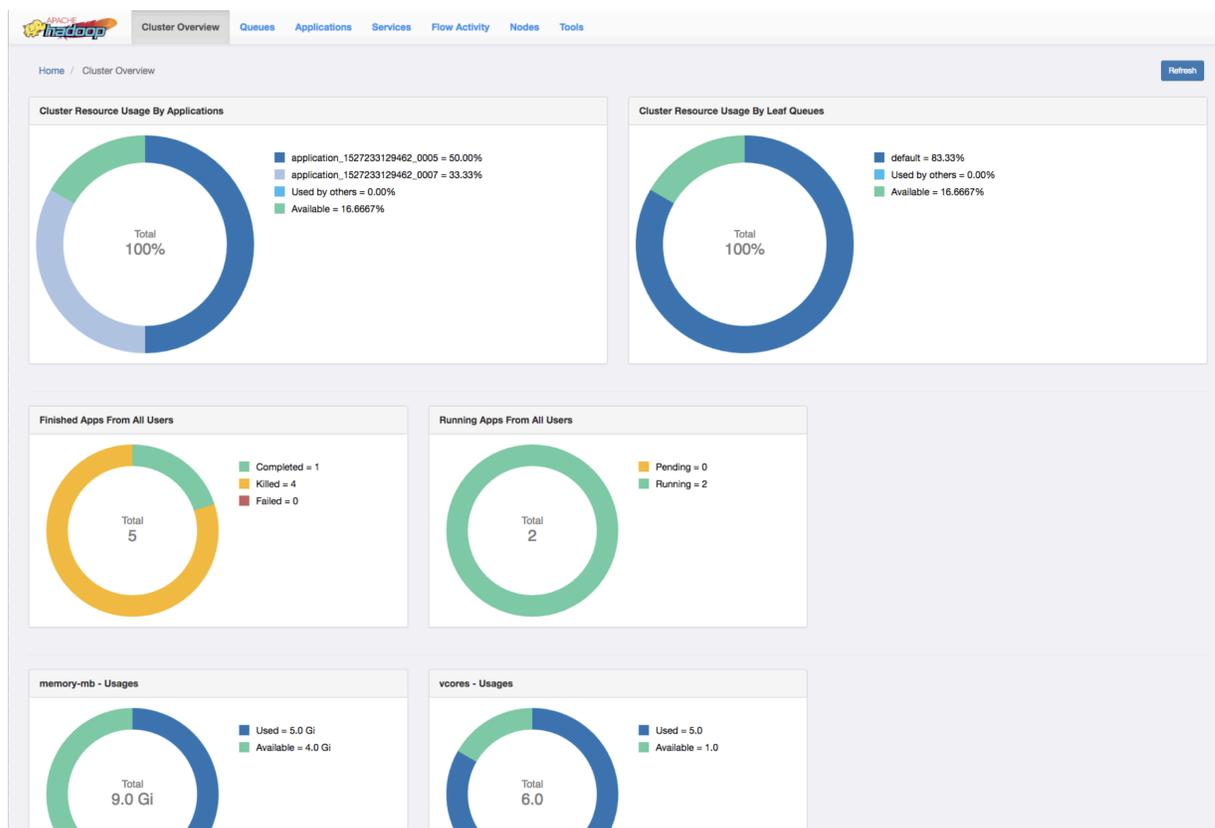
You can use the YARN Web User Interface to monitor clusters, queues, applications, services, and flow activities on CDP One.

Procedure

- Click Analyze YARN Jobs on the CDP One console to access the YARN Web UI.

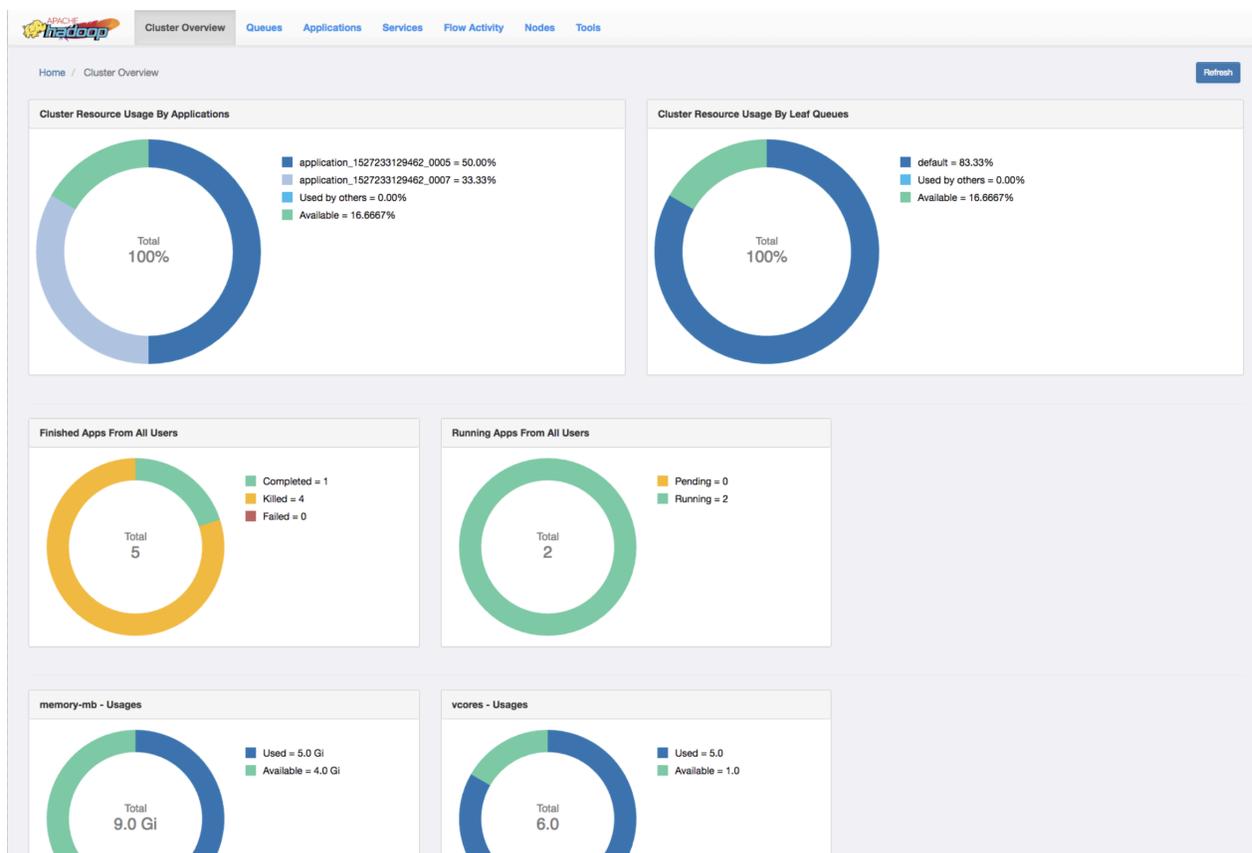


The YARN Cluster Overview page appears:



Viewing the Cluster Overview

The Cluster Overview page shows cluster resource usage by applications and queues, information about finished and running applications, and usage of memory and vCores in the cluster.



Cluster Resource Usage by Applications

Displays the percentage of cluster resources in use by applications and the percentage available for usage.

Cluster Resource Usage by Leaf Queues

Displays the percentage of cluster resources in use by leaf queues and the percentage available for usage.

Finished Apps From All Users

Displays the number of completed, killed, and failed applications.

Monitor Running Apps

Displays the number of pending and running applications.

memory-mb - Usages

Displays the amount of used and available memory.

vcores - Usages

Displays the number of used and available virtual cores.

Monitor Node Managers

Displays the status of the Node Managers under the following categories:

- Active
- Unhealthy
- Decommissioning
- Decommissioned

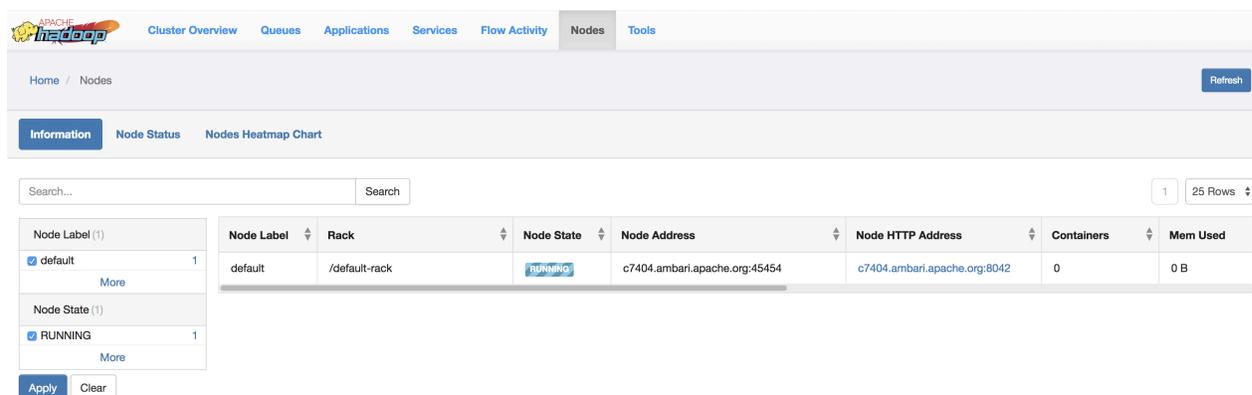
Viewing nodes and node details

The Nodes page on the YARN Web User Interface enables you to view information about the cluster nodes on which the NodeManagers are running.

The Nodes page displays details under the following headers: Information, Node Status and Nodes Heatmap Chart.

Information

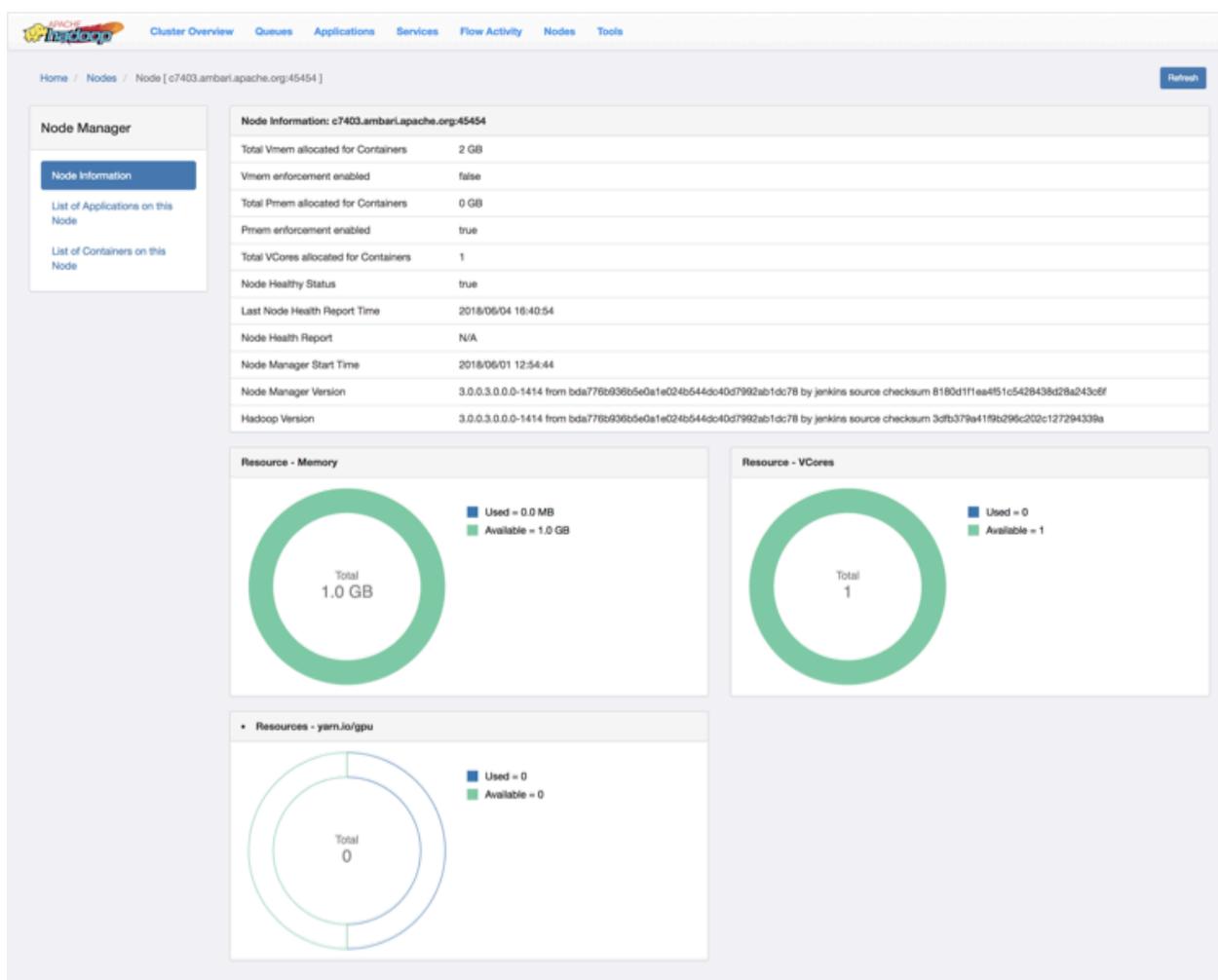
The Information tab displays the node details as shown in the following figure:



You can sort through any of the columns to view the details of the required nodes. You can also search to find the specific node labels from the entire list.

Node Manager page

To view the Node Manager page of any node label, click the corresponding Node HTTP address. The Node Manager page displays details of a node as shown in the following figure:



You can also view the resource usage in the following categories in a pie-chart representation:

1. Memory
2. VCores
3. yarn-io/GPU

Node status

This tab displays the node managers in a pictorial representation. It displays details such as the number of active nodes, number of unhealthy nodes, decommissioning nodes, and the number of decommissioned node managers.

Nodes heatmap chart

This tab graphically displays nodes on the basis of their usage of memory. You can enter host or rack details in the search bar to filter nodes.

Viewing queues and queue details

The Queues page is scheduler dependent and displays details of YARN queues. You can either view queues from all the partitions or filter to view queues of a partition.

Capacity Scheduler

APACHE HADOOP

Cluster Overview Queues Applications Services Flow Activity Nodes Tools

Home / Queues

capacity scheduler - Showing queues from all partitions

66.7% root

66.7% default

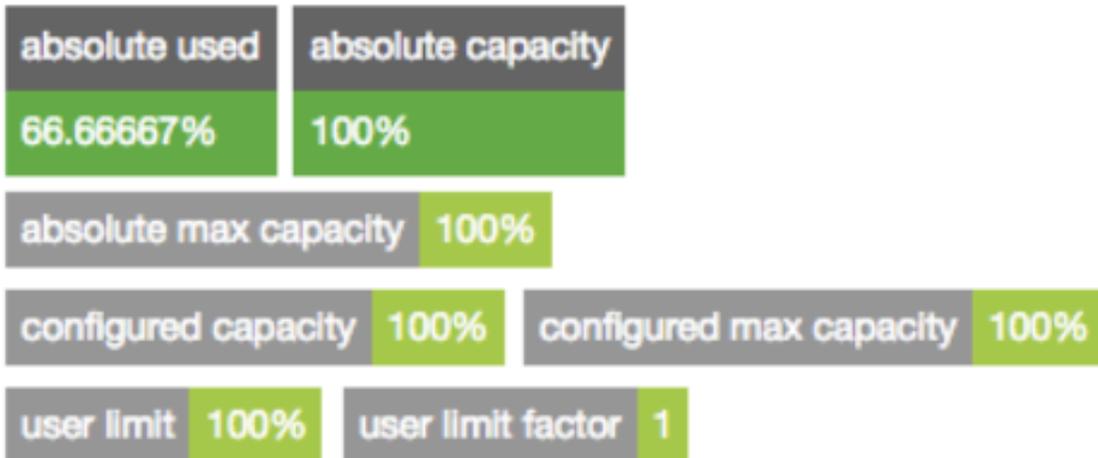
View queue details

In the capacity scheduler view, click the circle that represents a particular queue. The right column of the page gets updated with details of that queue.

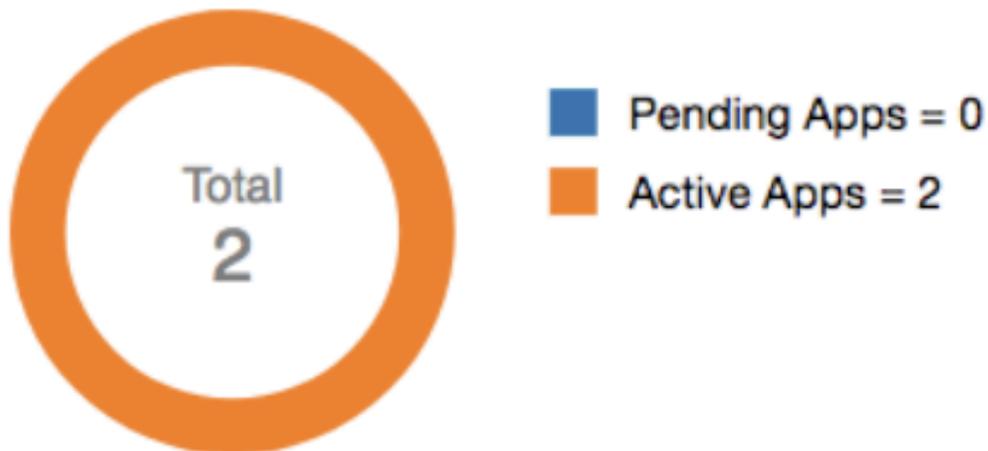
The following example shows the details of a queue:

default

● Running



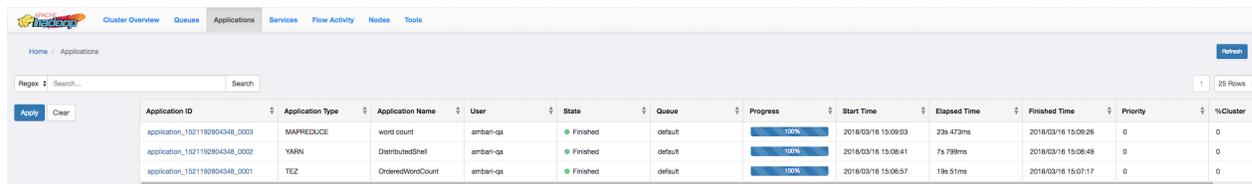
Running Apps From All Users in Queue



You can double-click the queue to view its details on a separate page. You can also view details of any application submitted to that queue by clicking its corresponding Application ID.

Viewing all applications

You can search for applications and view their details using the YARN Web User Interface.



The Applications page displays details of the YARN applications in a tabular form.

- Application ID: The identifier for the application.
- Application Type: Specifies the application type for Mapreduce, YARN, TEZ, or other applications.
- Application Name: Provides the name of the application
- User: The name of the user who is the owner of the application.
- State: The running state of the application.
- Queue: Specifies the name of the queue to which the application belongs.
- Progress: The progress of the application in a percentage display.
- Start Time: The time when an application run started.
- Elapsed Time: The time taken for the application run.
- Finished Time: The time of completion of the application run.
- Priority: The priority of running the application.
- %Cluster: The percentage of cluster resources used by the application run.

Searching applications

The Applications page displays the list of YARN applications in a tabular form. You can apply search filters on this list to show only those applications that match the search criteria.

About this task

You can specify the search criteria either as regular expressions or SQL statements.

Procedure

1. On the Applications page, select either Regex or SQL from the drop-down list depending on the type of search you want to perform.
2. In the Search box, specify the search criteria.

| Search Criteria | Description |
|-----------------|--|
| Regex | Lists the applications whose details match the search criterion specified as a regular expression. For example, if you want to view application runs longer than an hour, mention the regular expression <code>^hso</code> so that the YARN UI shows only those applications that mention the Elapsed Time value in terms of hours, minutes, and seconds. |
| SQL | Lists the applications whose details match the search criterion specified as a SQL statement. For example, if you want to view all the applications submitted by the user yarn, specify <code>"User"='yarn'</code> as the search criterion. |

3. Click Search to view details of the applications matching the search criteria.



Note:

- Apart from specifying search criteria to filter the list of applications, you can also select or clear the State and Queue check-boxes to view a specific set of applications depending on your requirements.
- You can sort the application entries in ascending or descending order by clicking the corresponding arrow next to any column header in the table.

Viewing application details

Clicking a YARN application on the Applications page displays its details.

You can view the following details for the selected application:

- Application Attempts
- Resource Usage
- Diagnostics
- Logs

The screenshot shows the Ambari interface for a YARN application named 'word count' with a status of 'SUCCEEDED'. The application ID is 'application_1527837750131_0002'. It is marked as 'Finished' and was completed by 'ambari-qa' on 2018/06/01 at 12:56:35. The interface includes tabs for 'Attempts List', 'Resource Usage', 'Diagnostics', and 'Logs'. The 'Attempts List' tab is active, showing a graph view of the application attempt. The graph shows a single attempt starting at 2018/06/01 12:56:00 and finishing at 2018/06/01 12:56:35. A detailed table for the attempt 'appattempt_1527837750131_0002_000001' is displayed on the right.

| appattempt_1527837750131_0002_000001 | |
|--------------------------------------|--|
| Application Attempt Id | appattempt_1527837750131_0002_000001 |
| Started Time | 2018/06/01 12:56:00 |
| Finished Time | 2018/06/01 12:56:35 |
| Elapsed Time | 35 Secs |
| AM Container Id | container_1527837750131_0002_01_000... |
| AM Node Id | c7403.ambari.apache.org:45454 |
| AM Node Web UI | c7403.ambari.apache.org:8042 |
| Log | Link |

Application Attempts

You can view the attempts in Graph View and Grid View.

Graph View

A graph displays the start time and finish time of the attempt. You can also find the details of the attempt such as application attempt ID, started time, finished time, elapsed time, AM Container ID, and AM Node ID in the form of a table. You can access the Node UI using the AM Node Web UI. You can also view the log by clicking on the log link.

Grid View

A table displays the details of the application attempts. You can find the details of the attempt such as application attempt ID, started time, finished time, elapsed time, AM Container ID, and AM Node ID. You can access the Node UI using the AM Node Web UI. You can also view the log by clicking on the log link.

Resource Usage

This tab displays the resources used by the application attempts.

Diagnostics

Use this tab to view the diagnostics details of the application attempts. You can view any outstanding resource requests, and scheduling information.

Logs

Use this tab to view logs specific to containers. Select an attempt from the dropdown list and select the specific container to view the desired logs.

UI Tools

You can view the YARN configuration and YARN Daemon logs on the Tools page.

YARN Configuration

You can see the values of the properties defined in the following configuration files:

1. Core Configuration: Details of properties defined in the core-default.xml file.
2. YARN Configuration: Details of properties defined in the yarn-default.xml file.
3. MapReduce Configuration: Details of the properties defined in the mapred-default.xml file.

YARN Daemon Logs

You can view the list of log files. Click on a log file to view its contents in another tab of your browser.

Using the YARN CLI to viewlogs for applications

Use the YARN CLI to view logs for running application. Configure the log aggregation to aggregate and write out logs for all containers belonging to a single Application grouped by NodeManagers to single log files at a configured location in the file system.

You can access container log files using the YARN ResourceManager web UI, but more options are available when you use the yarn logs CLI command.

View all log files for an application

Use the following command format to view all logs for an application:

```
yarn logs -applicationId <Application ID>
```

View a specific log type for an application

Use the following command format to view all logs of a particular type for an application:

```
yarn logs -applicationId <Application ID> -log_files <log_file_type>
```

For example, to view only the stderr error logs:

```
yarn logs -applicationId <Application ID> -log_files stderr
```

The -logFiles option also supports Java regular expressions, so the following format would return all types of log files:

```
yarn logs -applicationId <Application ID> -log_files .*
```

View ApplicationMaster log files

Use the following command format to view all ApplicationMaster container log files for an application:

```
yarn logs -applicationId <Application ID> -am ALL
```

Use the following command format to view only the first ApplicationMaster container log files:

```
yarn logs -applicationId <Application ID> -am 1
```

List container IDs

Use the following command format to list all container IDs for an application:

```
yarn logs -applicationId <Application ID> -show_application_log_info
```

View log files for one container

Once you have the container IDs, you can use the following command format to list the log files for a particular container:

```
yarn logs -applicationId <Application ID> -containerId <Container ID>
```

Show container log file information

Use the following command format to list all of the container log file names (types) for an application:

```
yarn logs -applicationId <Application ID> -show_container_log_info
```

You can then use the `-logFiles` option to view a particular log type.

View a portion of the log files for one container

For large container log files, you can use the following command format to list only a portion of the log files for a particular container:

```
yarn logs -applicationId <Application ID> -containerId <Container ID> -size <bytes>
```

To view the first 1000 bytes:

```
yarn logs -applicationId <Application ID> -containerId <Container ID> -size 1000
```

To view the last 1000 bytes:

```
yarn logs -applicationId <Application ID> -containerId <Container ID> -size -1000
```

Download logs for an application

Use the following command format to download logs to a local folder:

```
yarn logs -applicationId <Application ID> -out <path_to_local_folder>
```

The container log files are organized in parent folders labeled with the applicable node ID.

Display help for YARN logs

To display Help for yarn logs, run the following command:

```
yarn logs -help
```

Analyzing Spark jobs

You can use the Spark History Server web UI to analyze Spark jobs on CDP One. You can view application data for both completed and running Spark jobs.

The Spark History Server application UI includes the following information:

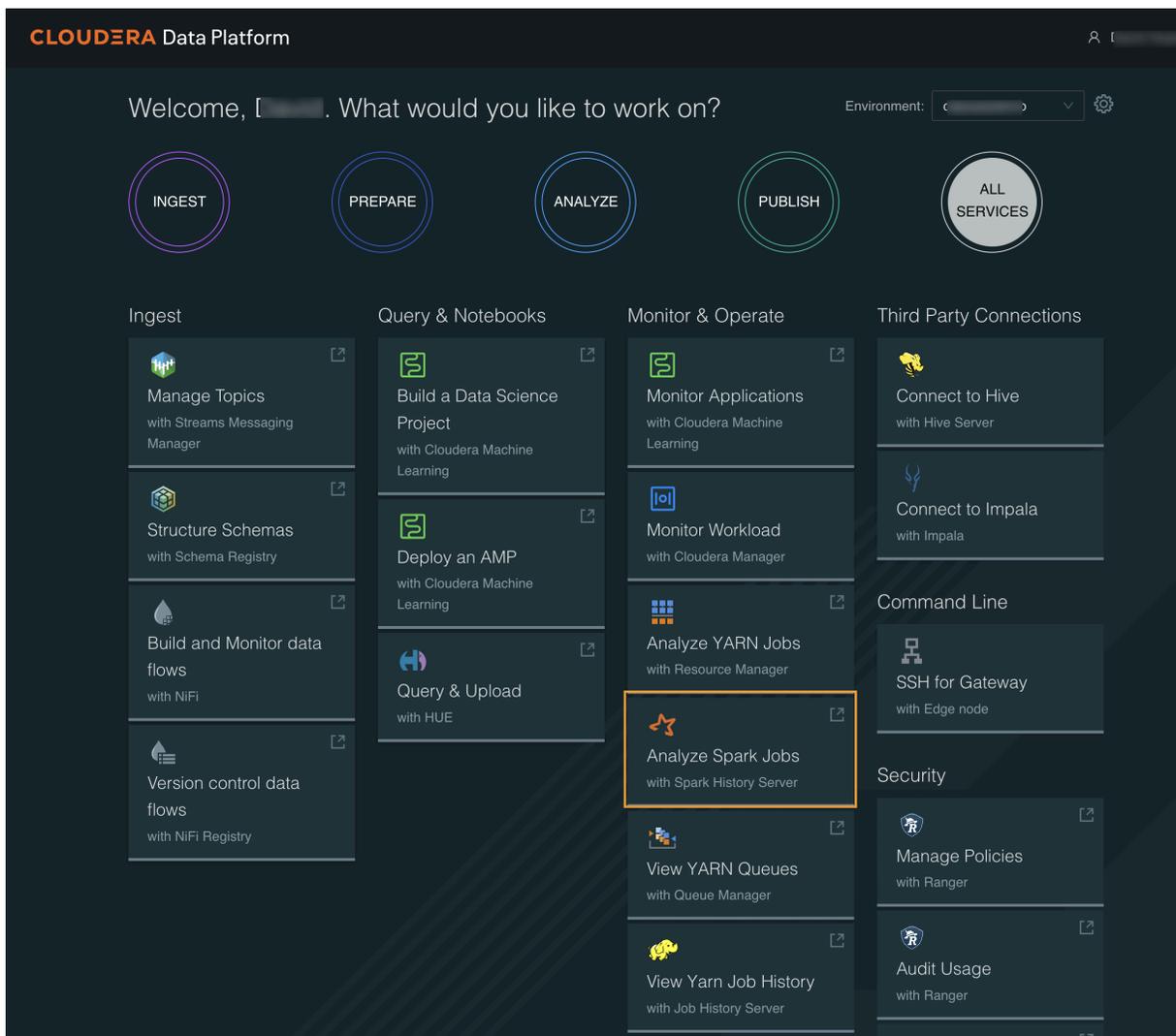
- An event timeline that displays the relative ordering and interleaving of application events. The timeline view is available on three levels: across all jobs, within one job, and within one stage. The timeline also shows executor allocation and deallocation.
- A list of stages and tasks.
- The execution directed acyclic graph (DAG) for each job.
- A summary of RDD sizes and memory usage.
- Environment - runtime information, property settings, library paths.
- Information about Spark SQL jobs.

Accessing the Spark History Server web UI

You can access the web application UI of a completed Spark application through the Spark history server.

Procedure

1. Click Analyze Spark Jobs on the CDP One console to access the Spark History Server Web UI.



2. In the list of applications, click an App ID link. The application UI appears.



Note: Completed applications are displayed by default. To display running applications, click Show incomplete applications.

Spark application data

Consider a job consisting of a set of transformation to join data from an accounts dataset with a weblogs dataset in order to determine the total number of web results for every account and then an action write the result to HDFS. In this example, the write is performed twice, resulting in two jobs. To view the application UI, in the History Server click the link in the App ID column:

Spark 1.6.0 History Server

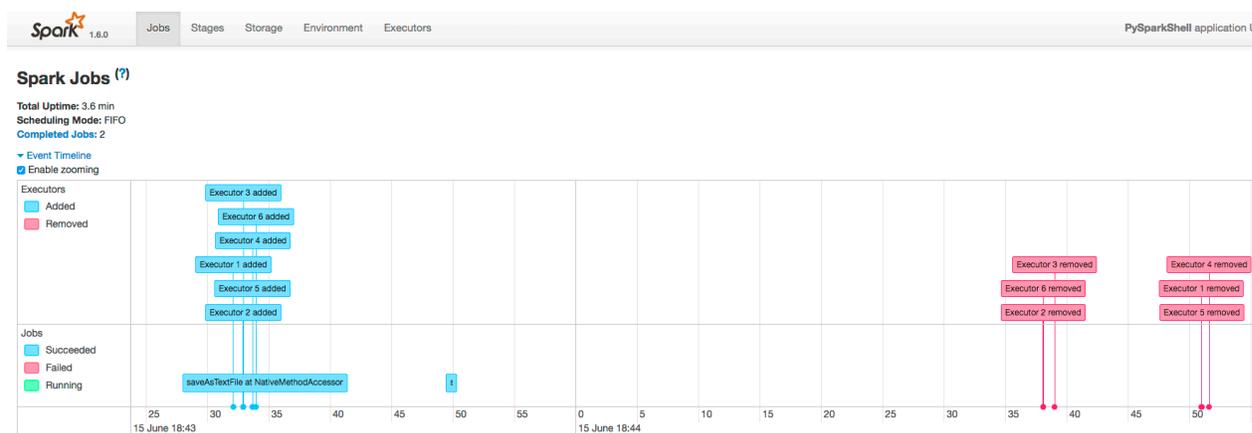
Event log directory: hdfs://vc0136.halixg.cloudera.com:8020/user/spark/applicationHistory

Showing 1-20 of 148

| App ID | App Name | Started | Completed | Duration | Spark User | Last Updated |
|--|--------------|---------------------|---------------------|----------|------------|---------------------|
| application_1463513516522_0731 | PySparkShell | 2016/06/15 18:41:54 | 2016/06/15 18:45:32 | 3.6 min | sparktest | 2016/06/15 18:45:32 |

The following screenshot shows the timeline of the events in the application including the jobs that were run and the allocation and deallocation of executors. Each job shows the last action, saveAsTextFile, run for the job. The timeline

shows that the application acquires executors over the course of running the first job. After the second job finishes, the executors become idle and are returned to the cluster.



Completed Jobs (2)

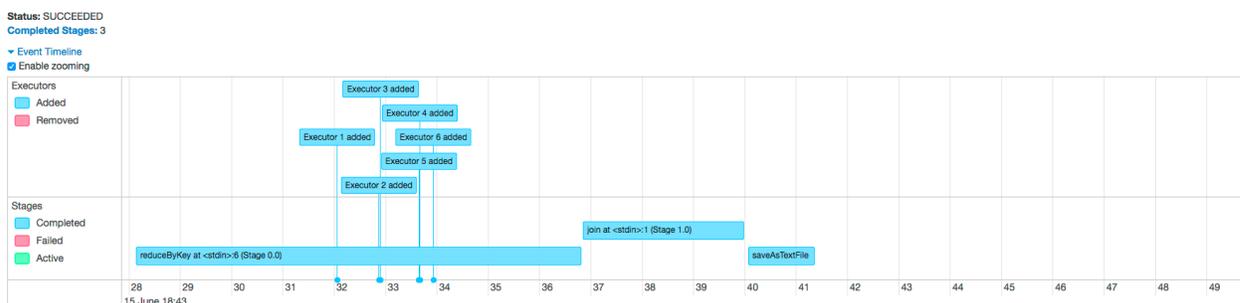
| Job Id | Description | Submitted | Duration | Stages: Succeeded/Total | Tasks (for all stages): Succeeded/Total |
|--------|--|---------------------|----------|-------------------------|---|
| 1 | saveAsTextFile at NativeMethodAccessorImpl.java:-2 | 2016/06/15 18:43:49 | 0.9 s | 1/1 (2 skipped) | 12/12 (18 skipped) |
| 0 | saveAsTextFile at NativeMethodAccessorImpl.java:-2 | 2016/06/15 18:43:27 | 13 s | 3/3 | 30/30 |

You can manipulate the timeline as follows:

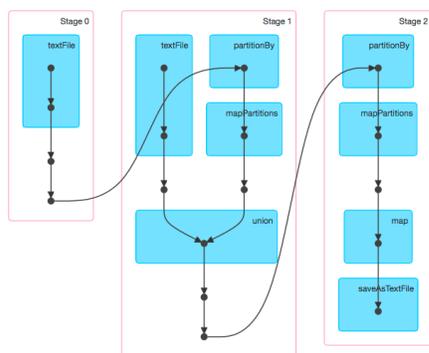
- Pan - Press and hold the left mouse button and swipe left and right.
- Zoom - Select the Enable zooming checkbox and scroll the mouse up and down.

To view the details for Job 0, click the link in the Description column. The following screenshot shows details of each stage in Job 0 and the DAG visualization. Zooming in shows finer detail for the segment from 28 to 42 seconds:

Details for Job 0



DAG Visualization



Completed Stages (3)

| Stage Id | Description | Submitted | Duration | Tasks: Succeeded/Total | Input | Output | Shuffle Read | Shuffle Write |
|----------|--|------------------------------|----------|------------------------|-------|---------|--------------|---------------|
| 2 | saveAsTextFile at NativeMethodAccessorImpl.java:-2 | +details 2016/06/15 18:43:40 | 1 s | 12/12 | | 55.3 KB | 3.1 MB | |
| 1 | join at <stdin>:1 | +details 2016/06/15 18:43:36 | 3 s | 12/12 | | 78.6 KB | | 3.1 MB |
| 0 | reduceByKey at <stdin>:6 | +details 2016/06/15 18:43:28 | 5 s | 6/6 | | | | 78.6 KB |

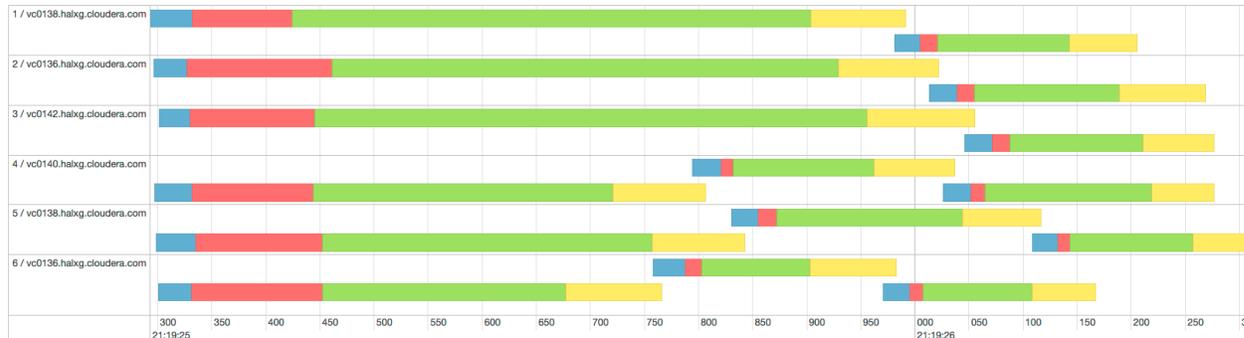
Clicking a stage shows further details and metrics:

Details for Stage 1 (Attempt 0)

Total Time Across All Tasks: 5 s
 Locality Level Summary: Node local: 9; Process local: 6
 Input Size / Records: 191.8 KB / 97324
 Shuffle Read: 78.6 KB / 216
 Shuffle Write: 2.8 MB / 1023

- ▶ DAG Visualization
- ▶ Show Additional Metrics
- ▶ Event Timeline
- ▶ Enable zooming

■ Scheduler Delay
 ■ Task Deserialization Time
 ■ Executor Computing Time
 ■ Getting Result Time
■ Shuffle Read Time
 ■ Shuffle Write Time
 ■ Result Serialization Time



Summary Metrics for 15 Completed Tasks

| Metric | Min | 25th percentile | Median | 75th percentile | Max |
|------------------------------|-------------|-----------------|--------------|-----------------|-----------------|
| Duration | 0.2 s | 0.2 s | 0.2 s | 0.4 s | 0.6 s |
| GC Time | 0 ms | 0 ms | 0 ms | 0 ms | 0 ms |
| Input Size / Records | 0.0 B / 0 | 0.0 B / 0 | 0.0 B / 1 | 0.0 B / 8187 | 63.9 KB / 24336 |
| Shuffle Read Size / Records | 0.0 B / 0 | 0.0 B / 0 | 0.0 B / 0 | 13.0 KB / 36 | 13.7 KB / 36 |
| Shuffle Write Size / Records | 299.0 B / 1 | 12.8 KB / 20 | 13.4 KB / 20 | 223.8 KB / 135 | 723.9 KB / 165 |

Aggregated Metrics by Executor

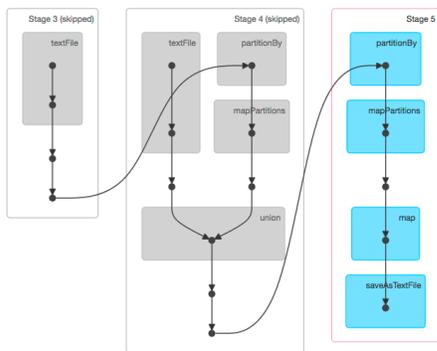
| Executor ID ▲ | Address | Task Time | Total Tasks | Failed Tasks | Succeeded Tasks | Input Size / Records | Shuffle Read Size / Records | Shuffle Write Size / Records |
|---------------|---------------------------------|-----------|-------------|--------------|-----------------|----------------------|-----------------------------|------------------------------|
| 1 | vc0138.halvg.cloudera.com:37289 | 0.9 s | 2 | 0 | 2 | 0.0 B / 24253 | 13.0 KB / 36 | 734.7 KB / 185 |
| 2 | vc0136.halvg.cloudera.com:43238 | 1.0 s | 2 | 0 | 2 | 0.0 B / 24336 | 13.0 KB / 36 | 737.0 KB / 185 |
| 3 | vc0142.halvg.cloudera.com:47133 | 1.0 s | 2 | 0 | 2 | 0.0 B / 24266 | 13.7 KB / 36 | 737.7 KB / 185 |
| 4 | vc0140.halvg.cloudera.com:59624 | 1 s | 3 | 0 | 3 | 127.9 KB / 18362 | 13.3 KB / 36 | 456.9 KB / 290 |
| 5 | vc0138.halvg.cloudera.com:33036 | 1 s | 3 | 0 | 3 | 63.9 KB / 8104 | 25.8 KB / 72 | 245.7 KB / 175 |
| 6 | vc0136.halvg.cloudera.com:56747 | 0.9 s | 3 | 0 | 3 | 0.0 B / 3 | 0.0 B / 0 | 900.0 B / 3 |

The web page for Job 1 shows how preceding stages are skipped because Spark retains the results from those stages:

Details for Job 1

Status: SUCCEEDED
 Completed Stages: 1
 Skipped Stages: 2

- ▶ Event Timeline
- ▶ DAG Visualization



Completed Stages (1)

| Stage Id | Description | Submitted | Duration | Tasks: Succeeded/Total | Input | Output | Shuffle Read | Shuffle Write |
|----------|--|------------------------------|----------|------------------------|-------|---------|--------------|---------------|
| 5 | saveAsTextFile at NativeMethodAccessorImpl.java:~2 | +details 2016/06/15 18:43:49 | 0.8 s | 12/12 | | 55.3 KB | 3.1 MB | |

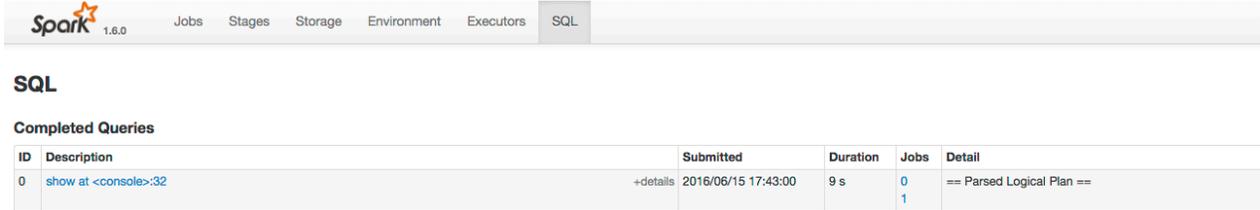
Skipped Stages (2)

| Stage Id | Description | Submitted | Duration | Tasks: Succeeded/Total | Input | Output | Shuffle Read | Shuffle Write |
|----------|--------------------------|------------------|----------|------------------------|-------|--------|--------------|---------------|
| 4 | join at <stdin>:1 | +details Unknown | Unknown | 0/12 | | | | |
| 3 | reduceByKey at <stdin>:6 | +details Unknown | Unknown | 0/6 | | | | |

Example Spark SQL Web Application

In addition to the screens described above, the web application UI of an application that uses the Spark SQL API also has an SQL tab. Consider an application that loads the contents of two tables into a pair of DataFrames, joins

the tables, and then shows the result. After you click the application ID, the SQL tab displays the final action in the query:



The screenshot shows the Spark 1.6.0 interface with the SQL tab selected. Below the navigation bar, the heading "SQL" is displayed. Underneath, the section "Completed Queries" contains a table with the following data:

| ID | Description | Submitted | Duration | Jobs | Detail |
|----|--|--|----------|--------|---------------------------|
| 0 | show at <console>:32 | +details 2016/06/15 17:43:00 | 9 s | 0 1 | == Parsed Logical Plan == |

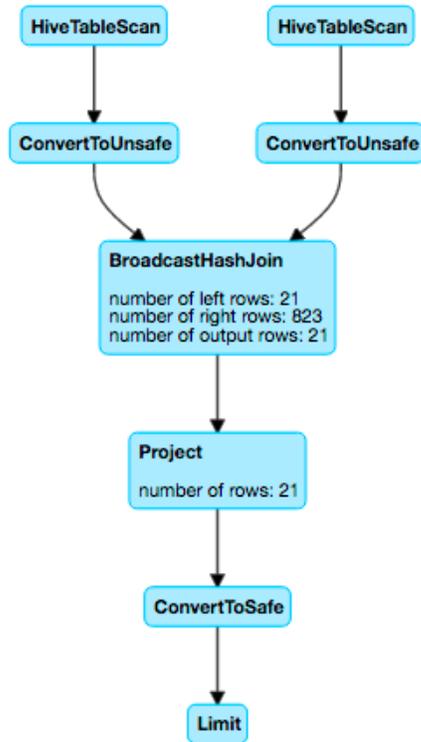
If you click the show link you see the DAG of the job. Clicking the Details link on this page displays the logical query plan:

Details for Query 0

Submitted Time: 2016/06/15 17:43:00

Duration: 9 s

Succeeded Jobs: 0 1



▼ Details

```

== Parsed Logical Plan ==
Limit 21
+- Project [code#0,description#1]
  +- Join Inner, Some((code#0 = code#4))
    :- Project [code#0,description#1,total_emp#2,salary#3]
      : +- MetastoreRelation default, sample_07, None
    +- Project [code#4,description#5,total_emp#6,salary#7]
      +- MetastoreRelation default, sample_08, None

== Analyzed Logical Plan ==
code: string, description: string
Limit 21
+- Project [code#0,description#1]
  +- Join Inner, Some((code#0 = code#4))
    :- Project [code#0,description#1,total_emp#2,salary#3]
      : +- MetastoreRelation default, sample_07, None
    +- Project [code#4,description#5,total_emp#6,salary#7]
      +- MetastoreRelation default, sample_08, None

== Optimized Logical Plan ==
Limit 21
+- Project [code#0,description#1]
  +- Join Inner, Some((code#0 = code#4))
    :- Project [code#0,description#1]
      : +- MetastoreRelation default, sample_07, None
    +- Project [code#4]
      +- MetastoreRelation default, sample_08, None

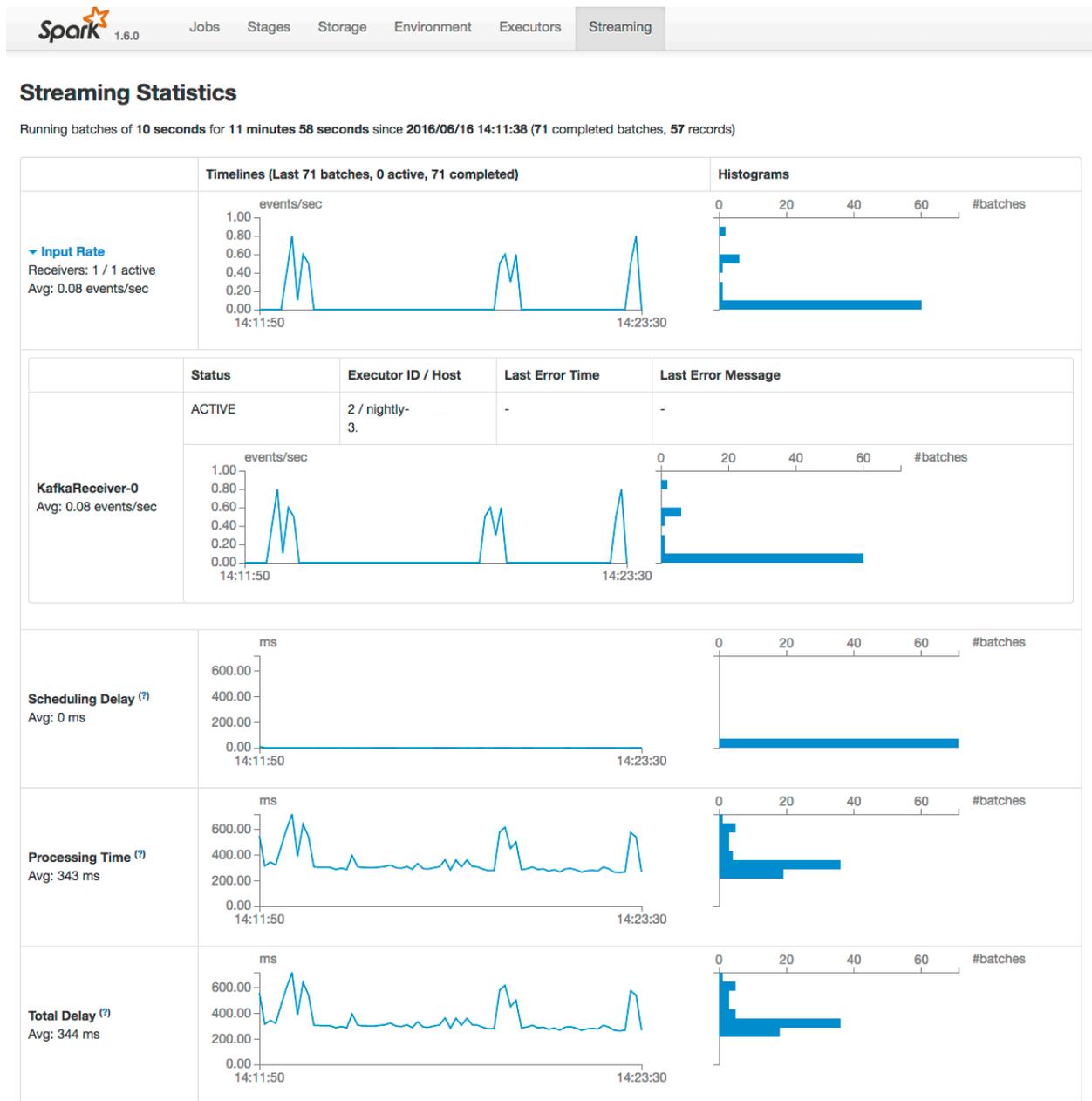
== Physical Plan ==
  
```

Example Spark Streaming Web Application



Note: The following example demonstrates the Spark driver web UI. Streaming information is not captured in the Spark History Server.

The Spark driver web application UI also supports displaying the behavior of streaming applications in the Streaming tab. If you run the example described in the Spark streaming example, and provide three bursts of data, the top of the tab displays a series of visualizations of the statistics summarizing the overall behavior of the streaming application:



The application has one receiver that processed 3 bursts of event batches, which can be observed in the events, processing time, and delay graphs. Further down the page you can view details of individual batches:

| Active Batches (0) | | | | | | |
|---------------------------------------|------------|---------------------------------|--------------------------------|-----------------------------|-----------------------------|--|
| Batch Time | Input Size | Scheduling Delay ^(?) | Processing Time ^(?) | Output Ops: Succeeded/Total | Status | |
| Completed Batches (last 71 out of 71) | | | | | | |
| Batch Time | Input Size | Scheduling Delay ^(?) | Processing Time ^(?) | Total Delay ^(?) | Output Ops: Succeeded/Total | |
| 2016/06/16 14:23:30 | 0 events | 1 ms | 0.3 s | 0.3 s | 1/1 | |
| 2016/06/16 14:23:20 | 8 events | 1 ms | 0.5 s | 0.5 s | 1/1 | |
| 2016/06/16 14:23:10 | 5 events | 1 ms | 0.6 s | 0.6 s | 1/1 | |
| 2016/06/16 14:23:00 | 0 events | 0 ms | 0.3 s | 0.3 s | 1/1 | |

To view the details of a specific batch, click a link in the Batch Time column. Clicking the 2016/06/16 14:23:20 link with 8 events in the batch, provides the following details:

Details of batch at 2016/06/16 14:23:20

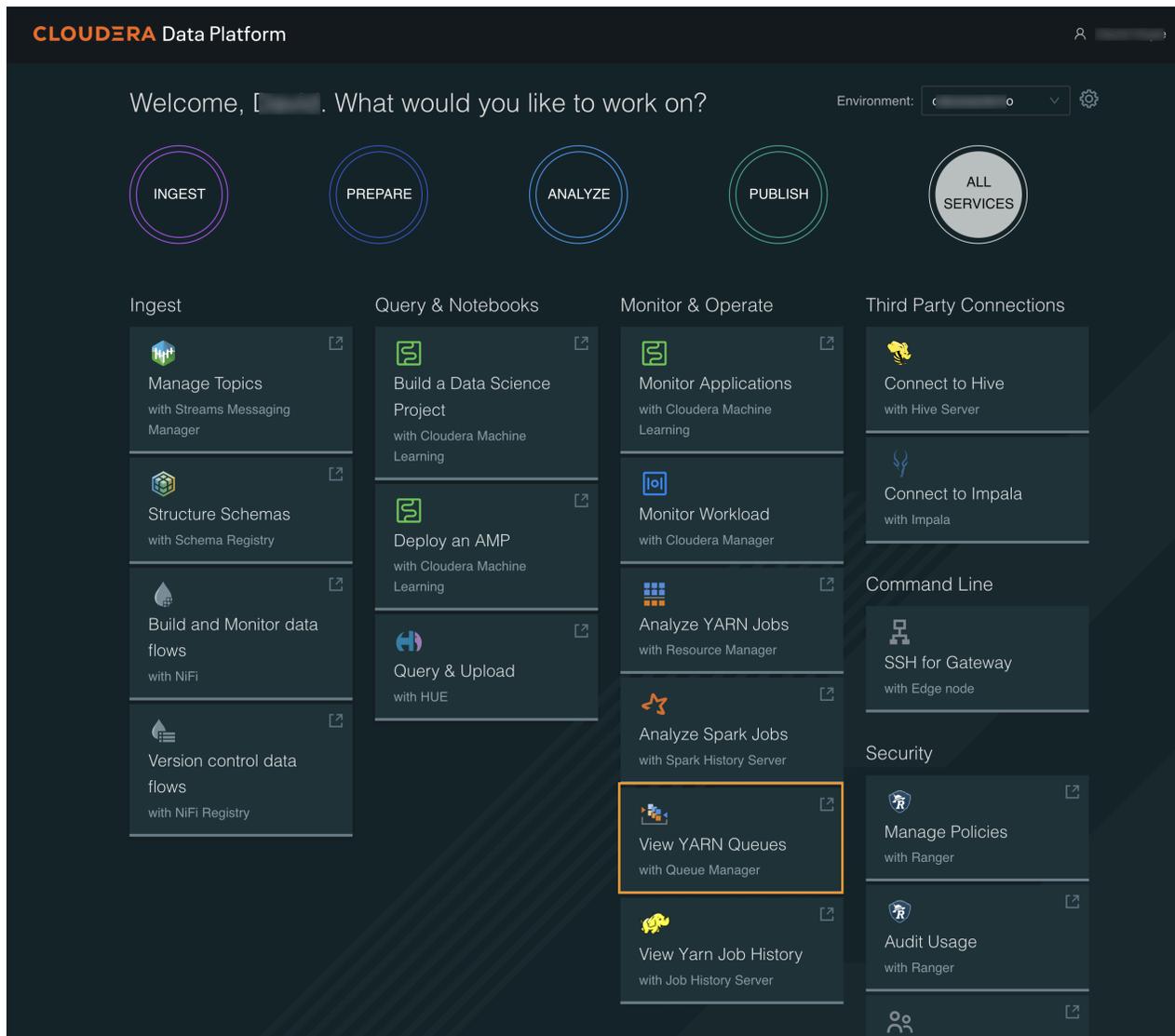
Batch Duration: 10 s
Input data size: 8 records
Scheduling delay: 1 ms
Processing time: 0.5 s
Total delay: 0.5 s

| Output Op Id | Description | Output Op Duration | Status | Job Id | Job Duration | Stages: Succeeded/Total | Tasks (for all stages): Succeeded/Total | Error |
|--------------|---|--------------------|-----------|--------|--------------|-------------------------|---|-------|
| 0 | callForeachRDD at NativeMethodAccessorImpl.java:-2 +details | 0.5 s | Succeeded | - | - | - | - | - |

Viewing YARN queues

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

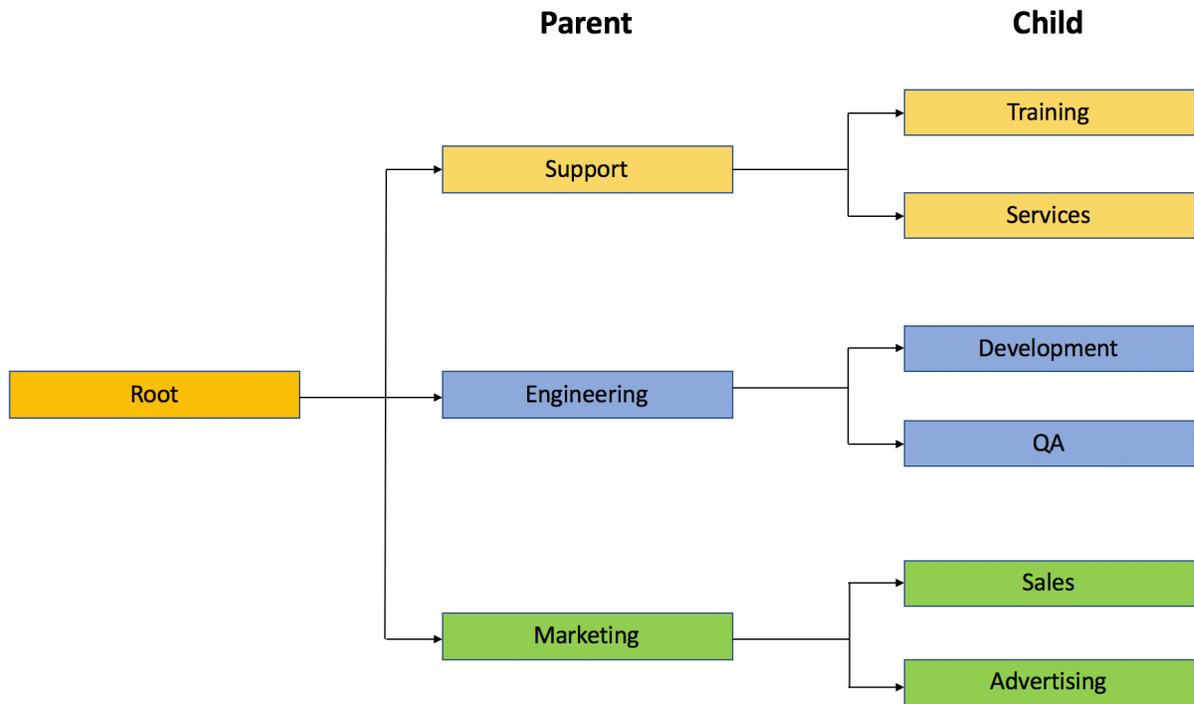
Click View YARN Queues on the CDP One console to access the YARN Queue Manager UI.



You can use the the YARN Queue Manager to view, sort, search, and filter queues. Queue Manager stores the history of previous changes and provides the ability to view the changes of each version in the Overview and Scheduler Configuration tabs. Previous versions are in read-only mode.

The fundamental unit of resource allocation 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 .

The screenshot shows the Cloudera Manager interface. On the left is a dark sidebar with the Cloudera Manager logo and a search bar. Below the search bar is a list of navigation items: Clusters, Hosts, Diagnostics, Audits, Charts, Replication, Administration, and Experiences (with a 'New' badge). The 'Clusters' item is highlighted with an orange border. The main content area is titled 'Cluster 1' and shows 'Cloudera Runtime 7.2.14 (Parcels)'. Below this, there is a list of services: HDFS-1, YARN Queue Manager, YARN-1, and ZOOKEEPER-1. To the right of this list is a menu with options: Hosts, Roles, Host Templates, Parcels, Send Diagnostic Data, Reports, Utilization Report, YARN Applications, YARN Queue Manager UI (highlighted with an orange border), and Static Service Pools. At the bottom of the main content area, there are links for 'Add Cluster' and 'Cloudera Management Service'.

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

The screenshot displays the Cloudera Manager interface for 'Cluster 1'. On the left is a dark sidebar with navigation options: Clusters, Hosts, Diagnostics, Audits, Charts, Replication, Administration, Experiences (marked as 'New'), and Parcels. The main content area shows the 'Overview' tab selected, with a 'Partition' dropdown set to 'default'. Below this is a 'Level 1' queue tree. A queue named 'root' is highlighted with a 100% completion status and a green checkmark. A context menu is open over the 'root' queue, listing the following actions: 'Edit Queue Properties', 'Edit Child Queues', 'Add Child Queue' (highlighted in blue), and 'More Information'. A second queue is partially visible to the right of 'root', showing a 90% completion status.

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.

root
✕

Memory: 16,384 MiB; vCores: 16

Memory
vCores ⓘ

| CONFIGURED MEMORY | MAXIMUM MEMORY | |
|---|---|--|
| <input style="width: 80px;" type="text" value="16384"/> MiB | <input style="width: 80px;" type="text" value="16384"/> MiB | ✔ default Memory: 16,384 MiB; vCores: 16 |
| <input style="width: 80px;" type="text" value="5000"/> MiB | <input style="width: 80px;" type="text" value="16384"/> MiB | <input style="width: 100%; height: 20px;" type="text" value="Support"/> |

ⓘ Additional queue properties can be set later by clicking **View/Edit Queue Properties** on the new queue's menu.

Total Configured Memory: 21,384 MiB, Delta to be adjusted: -5,000 MiB

Total Configured vCores: 20, Delta to be adjusted: -4

- 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.

root

Memory: 16.0 GiB; vCores: 16

| CONFIGURED CAPACITY | MAXIMUM CAPACITY | |
|-----------------------------------|------------------------------------|---|
| <input type="text" value="40"/> % | <input type="text" value="100"/> % | ✓ default Memory: 16.0 GiB; vCores: 16 |
| <input type="text" value="60"/> % | <input type="text" value="100"/> % | <input type="text" value="Support"/> |

i Additional queue properties can be set later by clicking **View/Edit Queue Properties** on the new queue's menu.

Total Configured Capacity: 100%

Cancel

Save

- 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.

root
✕

Memory: 28.0 GiB; vCores: 8

| CONFIGURED WEIGHT | MAXIMUM CAPACITY | |
|-------------------|------------------|---|
| 1 w | 100 % | ✔ default Memory: 28.0 GiB; vCores: 8 |
| 1 w | 100 % | <input style="width: 100%;" type="text" value="Development"/> |

i Additional queue properties can be set later by clicking **View/Edit Queue Properties** on the new queue's menu.

4. Click Save.

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

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.

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:

Accessible Cluster Capacity ×

Memory: 36,864 MiB; vCores: 16

Memory
vCores ⓘ

| CONFIGURED MEMORY | MAXIMUM MEMORY | |
|-------------------|----------------|---|
| 28672 MiB | 28672 MiB | <input checked="" type="checkbox"/> root Memory: 28,672 MiB; vCores: 8 |

Total Configured Memory: 28,672 MiB, Delta to be adjusted: +8,192 MiB

Total Configured vCores: 8, Delta to be adjusted: +8

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

Accessible Cluster Capacity ✕

Memory: 8,192 MiB; vCores: 8

Memory vCores ⓘ

| CONFIGURED MEMORY | MAXIMUM MEMORY | |
|-------------------|----------------|--|
| 36864 MiB | 36864 MiB | <input checked="" type="checkbox"/> root Memory: 36,864 MiB; vCores: 16 |

Maximum memory must not exceed 8,192 MiB

Total Configured Memory: 36,864 MiB, Delta to be adjusted: -28,672 MiB
 Total Configured vCores: 16, Delta to be adjusted: -8

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.

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.



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.

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.

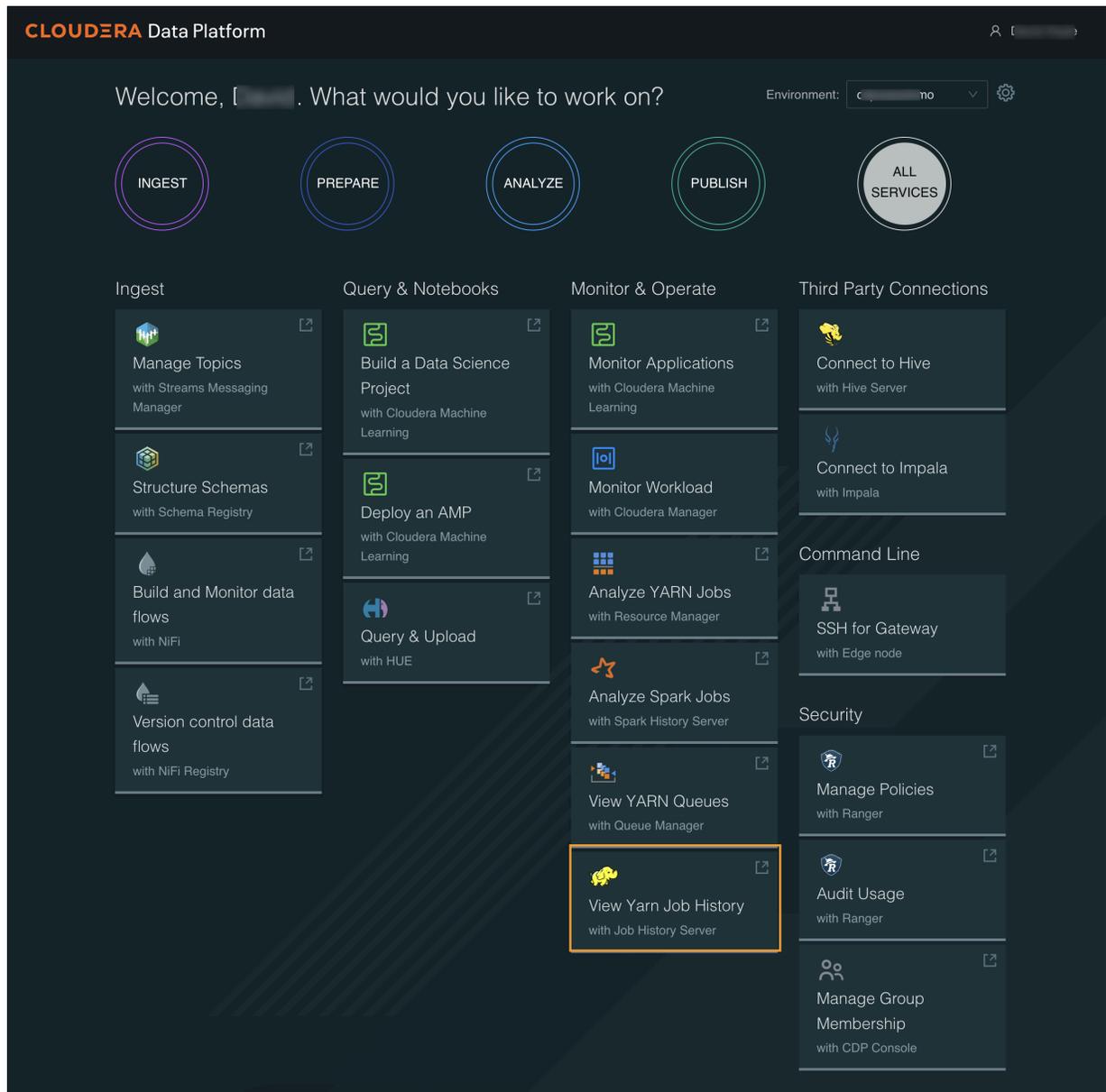
6. Click Save.

Viewing the YARN job history

You can use the YARN Job History Server UI to view information about YARN jobs on CDP One, such as start and end time, job ID, name, user, queue, and whether or not the job succeeded.

Procedure

- Click View YARN Job History on the CDP One console to access the YARN Job History Server UI.



The YARN Job History page appears:



JobHistory Logged in as: dhoyle

Application

- About Jobs
- Tools

Retired Jobs

Show 20 entries

| Submit Time | Start Time | Finish Time | Job ID | Name | User | Queue | State | Maps Total | Maps Completed | Reduces Total | Reduces Completed | Elapsed Time |
|-------------------------|-------------------------|-------------------------|------------------------|-----------|------|---------|-----------|------------|----------------|---------------|-------------------|----------------------|
| 2022.07.31 18:36:06 UTC | 2022.07.31 19:21:16 UTC | 2022.07.31 19:21:34 UTC | job_1657577265389_5147 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 17sec |
| 2022.07.31 18:35:00 UTC | 2022.07.31 19:20:46 UTC | 2022.07.31 19:21:01 UTC | job_1657577265389_5146 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 15sec |
| 2022.07.31 18:34:53 UTC | 2022.07.31 19:20:14 UTC | 2022.07.31 19:20:30 UTC | job_1657577265389_5145 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 16sec |
| 2022.07.31 18:34:11 UTC | 2022.07.31 19:19:40 UTC | 2022.07.31 19:19:58 UTC | job_1657577265389_5144 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 17sec |
| 2022.07.31 18:33:46 UTC | 2022.07.31 19:19:07 UTC | 2022.07.31 19:19:25 UTC | job_1657577265389_5143 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 18sec |
| 2022.07.31 16:49:23 UTC | 2022.07.31 18:35:08 UTC | 2022.07.31 18:35:30 UTC | job_1657577265389_5141 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 21sec |
| 2022.07.31 16:48:31 UTC | 2022.07.31 18:34:23 UTC | 2022.07.31 18:34:50 UTC | job_1657577265389_5139 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 26sec |
| 2022.07.31 16:48:23 UTC | 2022.07.31 18:33:44 UTC | 2022.07.31 18:34:08 UTC | job_1657577265389_5138 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 23sec |
| 2022.07.31 16:34:28 UTC | 2022.07.31 16:48:38 UTC | 2022.07.31 16:49:01 UTC | job_1657577265389_5134 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 23sec |
| 2022.07.31 16:33:29 UTC | 2022.07.31 16:47:49 UTC | 2022.07.31 16:48:07 UTC | job_1657577265389_5131 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 17sec |
| 2022.07.31 16:30:15 UTC | 2022.07.31 16:34:26 UTC | 2022.07.31 16:34:42 UTC | job_1657577265389_5129 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 16sec |
| 2022.07.31 16:29:24 UTC | 2022.07.31 16:33:52 UTC | 2022.07.31 16:34:10 UTC | job_1657577265389_5128 | [blurred] | hive | default | SUCCEEDED | 1 | 1 | 0 | 0 | 00hrs, 00mins, 18sec |

- You can use the Search box to search for jobs.
- Click a column header to sort the jobs list in ascending or descending order based on the applicable column.