Cloudera Runtime 7.2.18

# Monitoring Apache Impala

**Date published: 2020-11-30**
**Date modified: 2024-03-25**

CLOUDERA

**https://docs.cloudera.com/**

# Legal Notice

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

# Impala Logs

You can review Impala log files on each host, when you have traced an issue back to a specific system. The Impala logs contain information about any errors Impala encountered, jobs Impala has completed, and settings Impala is configured with.

The logs store information about Impala startup options. This information appears once for each time Impala is started and may include:

- Machine name.
- Impala version number.
- Flags used to start Impala.
- CPU information.
- The number of available disks.

A new set of log files is produced each time the associated daemon is restarted. These log files have long names including a timestamp. The .INFO, .WARNING, and .ERROR files are physically represented as symbolic links to the latest applicable log files.

Review Impala log files on each host, when you have traced an issue back to a specific system:

- By using the web interface at http://host-name:25000/logs where host-name is your Cloudera cluster host name.

  The web interface limits the amount of logging information displayed. To view every log entry, access the log files directly through the file system. Impala log files can often be several megabytes in size.
- By examining the contents of the log file

  By default, the Impala logs are stored at /var/log/impalad/, /var/log/catalogd/, and /var/log/statestore/. The most comprehensive log, showing informational, warning, and error messages, is in the file name impalad.INFO.

  For each of impalad, statestored, catalogd:

  - Examine the .INFO files to see configuration settings for the processes.
  - Examine the .WARNING files to see all kinds of problem information, including such things as suboptimal settings and also serious runtime errors.
  - Examine the .ERROR and/or .FATAL files to see only the most serious errors, if the processes crash, or queries fail to complete. These messages are also in the .WARNING file.

Cloudera Manager collects front-end and back-end logs together into a single view and let you do a search across log data for all the managed nodes in  Diagnostics Logs .

## Managing Logs

You must configure the Impala log settings to change the default log locations, to rotate logs, or to log verbose levels.

### Procedure

1. To change log file locations:
   a) In Cloudera Manager, navigate to  Impala service Configuration .
   b) In the search field, type log_dir.
   c) Specify the new log directories for Impala Daemon, Catalog Server, or StateStore in the respective fields:
      - Impala Daemon Log Directory
      - Catalog Server Log Directory
      - StateStore Log Directory
   d) Click Save Changes and restart Impala.

2. To set up log rotation:
   a) In Cloudera Manager, navigate to  Impala service Configuration .
   b) In the search field, type max_log_files.
   c) Specify the values for Impala Daemon, Catalog Server, or StateStore in the respective fields:
      - Impala Daemon Maximum Log Files
      - Catalog Server Maximum Log Files
      - StateStore Maximum Log Files
   d) Click Save Changes and restart Impala.

   The above configuration option specifies how many log files to keep at each severity level (INFO, WARNING, ERROR, and FATAL).

   - A value of 0 preserves all log files, in which case you would set up set up manual log rotation using your Linux tool or technique of choice.
   - A value of 1 preserves only the very latest log file.
   - The default value is 10.

   For some log levels, Impala logs are first temporarily buffered in memory and only written to disk periodically. The --logbufsecs setting controls the maximum time that log messages are buffered for. For example, with the default value of 5 seconds, there may be up to a 5 second delay before a logged message shows up in the log file.

   It is not recommended that you set --logbufsecs to 0 as the setting makes the Impala daemon to spin in the thread that tries to delete old log files.
3. To specify how often the log information is written to disk:
   a) In Cloudera Manager, navigate to  Impala service Configuration .
   b) In the search field, type logbuflevel.
   c) Specify the values for Impala Daemon, Catalog Server, or StateStore in the respective fields:
      - Impala Daemon Log Buffer Level
      - Catalog Server Log Buffer Level
      - StateStore Log Buffer Level

      The default is 0, meaning that the log is immediately flushed to disk when Impala outputs an important messages such as a warning or an error, but less important messages such as informational ones are buffered in memory rather than being flushed to disk immediately.
   d) Click Save Changes and restart Impala.

**4.** To set the logging verbose levels:

a) In Cloudera Manager, navigate to Impala service Configuration .

b) In the search field, type GLOG_v.

c) Specify the values for Impala Daemon, Catalog Server, or StateStore in the respective fields:

- Impala Daemon Verbose Log Level
- Catalog Server Verbose Log Level
- StateStore Verbose Log Level

d) Click Save Changes and restart Impala.

As logging levels increase, the categories of information logged are cumulative. For example, GLOG_v=2 records everything GLOG_v=1 records, as well as additional information.

Increasing logging levels imposes performance overhead and increases log size. Cloudera recommends using GLOG_v=1 for most cases: this level has minimal performance impact but still captures useful troubleshooting information.

Additional information logged at each level of GLOG_v is as follows:

- 1: The default level. Logs information about each connection and query that is initiated to an impalad instance, including runtime profiles.
- 2: Everything from the previous level plus information for each RPC initiated. This level also records query execution progress information, including details on each file that is read.
- 3: Everything from the previous level plus logging of every row that is read. This level is only applicable for the most serious troubleshooting and tuning scenarios, because it can produce exceptionally large and detailed log files, potentially leading to its own set of performance and capacity problems.

**Note:** For performance reasons, Cloudera highly recommends not setting the most verbose logging level to 3.

# Impala lineage

You can use the Atlas lineage graph to understand the source and impact of data and changes to data over time and across all your data.

Atlas collects metadata from Impala to represent the lineage among data assets. The Atlas lineage graph shows the input and output processes that the current entity participated in. Entities are included if they were inputs to processes that lead to the current entity or they are output from processes for which the current entity was an input. Impala processes follow this pattern.

Note that lineage is not updated between a table and views that the table is a part of when an Impala ALTER TABLE operation runs on the table.

**Related Information**

Impala metadata collection

# Web User Interface for Debugging

You can use the Impala daemons (`impalad`, `statestored`, and `catalogd`) Web UI to display the diagnostic and status information. Each of these Impala daemons includes a built-in web server.

**Impala Daemon Web UI**

The Impala Daemon (`impalad`) Web UI includes information about configuration settings, running and completed queries, and associated performance and resource usage for queries. In particular, the Details link for each query displays alternative views of the query including a graphical representation of the plan, and the output of the EXPLAIN, SUMMARY, and PROFILE statements from `impala-shell`. Each host that runs the `impalad` daemon has its own instance of the Web UI, with details about those queries for which that host served as the coordinator. The `impalad` Web UI is primarily used for diagnosing query problems that can be traced to a particular node.

**StateStore Web UI**

The StateStore (`statestored`) Web UI includes information about memory usage, configuration settings, and ongoing health checks performed by statestored. Because there is only a single instance of the statestored within any Impala cluster, you access the Web UI only on the particular host that serves as the Impala StateStore.

**Catalog Server Web UI**

The Catalog Server (`catalogd`0 Web UI includes information about the databases, tables, and other objects managed by Impala, in addition to the resource usage and configuration settings of the catalogd. Because there is only a single instance of the catalogd within any Impala cluster, you access the Web UI only on the particular host that serves as the Impala Catalog Server.

## Debug Web UI for Impala Daemon

You can use the Impala Daemon (`impalad`) Web UI to view information about configuration settings, running and completed queries, and associated performance and resource usage for queries.

To debug and troubleshoot an `impalad` using a web-based interface, open the URL http ://*IMPALA#SERVER#HOSTNAME*:25000/ in a browser. (For secure clusters, use the prefix https:// instead of http://.)

Because each Impala node produces its own set of debug information, you should choose a specific node that you want to investigate an issue on.

**Main Page**

The main impalad Web UI page at / lists the following information about the impalad:

- The version of the `impalad` daemon

  The Version section also contains other information, such as when Impala was built and what build flags were used.
- Process start time
- Hardware information
- OS information
- Process information
- CGroup information

**Admission Controller Page**

The Admission Controller impalad debug Web UI is at /admission page under the main impalad Web UI.

Use the /admission page to troubleshoot queued queries and the admission control.

The admission page provides the following information about each resource pool to which queries have been submitted at least once:

- Time since the statestored received the last update
- A warning if this impalad is considered disconnected from the statestored and thus the information on this page could be stale.
- Pool configuration
- Queued queries submitted to this coordinator, in the order of submission
- Running queries submitted to this coordinator

- Pool stats
  - Average of time in queue: An exponential moving average which represents the average time in queue over the last 10 to 12 queries. If a query is admitted immediately, the wait time of 0 is used in calculating this average wait time.
- Histogram of the distribution of peak memory used by queries admitted to the pool

  Use the histogram to figure out settings for the minimum and maximum query MEM_LIMIT ranges for this pool.

  The histogram displays data for all queries admitted to the pool, including the queries that finished, got canceled, or encountered an error.

Click on the pool name to only display information relevant to that pool. You can then refresh the debug page to see only the information for that specific pool.

Click Reset informational stats for all pools to reset the stats that keep track of historical data, such as Totals stats, Time in queue (exponential moving average), and the histogram.

The above information is also available as a JSON object from the following HTTP endpoint:

```
http://IMPALA-SERVER-HOSTNAME:PORT/admission?json
```

### Known Backends Page

The Known backends page of the impalad debug Web UI is at /backends under the main impalad Web UI.

This page lists the following info for each of the `impalad` nodes in the cluster. Because each `impalad` daemon knows about every other `impalad` daemon through the StateStore, this information should be the same regardless of which node you select.

- Address of the node: Host name and port
- KRPC address: The KRPC address of the impalad. Use this address when you issue the SHUT DOWN command for this impalad.
- Whether acting as a coordinator
- Whether acting as an executor
- Quiescing status: Specify whether the graceful shutdown process has been initiated on this impalad.
- Memory limit for admission: The amount of memory that can be admitted to this backend by the admission controller.
- Memory reserved: The amount of memory reserved by queries that are active, either currently executing or finished but not yet closed, on this backend.

  The memory reserved for a query that is currently executing is its memory limit, if set. Otherwise, if the query has no limit or if the query finished executing, the current consumption is used.
- Memory of the queries admitted to this coordinator: The memory submitted to this particular host by the queries admitted by this coordinator.

### Catalog Page

The Catalog page of the impalad debug Web UI is at /catalog under the main impalad Web UI.

This page displays a list of databases and associated tables recognized by this instance of `impalad`. You can use this page to locate which database a table is in, check the exact spelling of a database or table name, look for identical table names in multiple databases. The primary debugging use case would be to check if an impalad instance has knowledge of a particular table that someone expects to be in a particular database.

### Hadoop Configuration

The Hadoop Configuration page of the impalad debug Web UI is at /hadoop-varz under the main impalad Web UI.

This page displays the Hadoop common configurations that Impala is running with.

### JMX

The JMX page of the impalad debug Web UI is at /jmx under the main impalad Web UI.

This page displays monitoring information about various JVM subsystems, such as memory pools, thread management, runtime. etc.

### Java Log Level

The Change log level page of the impalad debug Web UI is at /log_level under the main impalad Web UI.

This page displays the current Java and backend log levels, and it allows you to change the log levels dynamically without having to restart the impalad.

### Logs Page

The INFO logs page of the impalad debug Web UI is at /logs under the main impalad Web UI.

This page shows the last portion of the impalad.INFO log file, including the info, warning, and error logs for the `impalad`. You can see the details of the most recent operations, whether the operations succeeded or encountered errors. This page provides one central place for the log files and saves you from looking around the filesystem for the log files, which could be in different locations on clusters that use cluster management software.

### Memz Page

The Memory Usage page of the impalad debug Web UI is at /memz under the main impalad Web UI.

This page displays the summary and detailed information about memory usage by the impalad.

### Metrics Page

The Metrics page of the impalad debug Web UI is at /metrics under the main impalad Web UI.

This page displays the current set of metrics, counters and flags representing various aspects of impalad internal operations.

### Queries Page

The Queries page of the impalad debug Web UI is at /queries under the main impalad Web UI.

This page lists:

- Currently running queries
- Queries that have completed their execution, but have not been closed yet
- Completed queries whose details still reside in memory

The queries are listed in reverse chronological order, with the most recent at the top. You can control the amount of memory devoted to completed queries by specifying the -#-#query_log_size startup option for impalad.

This page provides:

- How many SQL statements are failing (State value of EXCEPTION)
- How large the result sets are (# rows fetched)
- How long each statement took (Start Time and End Time)

Click the Details link for a query to display the detailed performance characteristics of that query, such as the profile output.

On the query detail page, in the Profile tab, you have options to export the query profile output to the Thrift, text, or Json format.

The Queries page also includes the Query Locations section that lists the number of running queries with fragments on this host.

### RPC Services Page

The RPC durations page of the impalad debug Web UI is at /rpcz under the main impalad Web UI.

This page displays information, such as the duration, about the RPC communications of this impa lad with other Impala daemons.

### Sessions Page

The Sessions page of the impalad debug Web UI is at /session under the main impalad Web UI.

This page displays information about the sessions currently connected to this `impalad` instance. For example, sessions could include connections from the impala-shell command, JDBC or ODBC applications, or the Impala Query UI in the Hue web interface.

### Threadz Page

The Threads page of the impalad debug Web UI is at /threadz under the main impalad Web UI.

This page displays information about the threads used by this instance of impalad, and it shows which categories they are grouped into. Making use of this information requires substantial knowledge about Impala internals.

### Varz Page

The Varz page of the impalad debug Web UI is at /varz under the main impalad Web UI.

This page shows the configuration settings in effect when this instance of impalad communicates with other Hadoop components such as HDFS and YARN. These settings are collected from a set of configuration files.

The bottom of this page also lists all the command-line settings in effect for this instance of impa lad.

### Prometheus Metrics Page

At /metrics_prometheus, under the main impalad Web UI, the metrics are generated in Prometheus exposition format that Prometheus can consume for event monitoring and alerting.

The /metrics_prometheus is not shown in the Web UI list of pages.

## Debug Web UI for StateStore

You can use the StateStore (`statestored`) Web UI to view information about memory usage, configuration settings, and ongoing health checks performed by statestored.

To debug and troubleshoot the statestored daemon using a web-based interface, open the URL http ://*IMPALA#SERVER#HOSTNAME*:25010/ in a browser. (For secure clusters, use the prefix https:// instead of http://.)

### Main Page

The main statestored Web UI page at / lists the following information about the statestored:

- The version of the statestored daemon
- Process start time
- Hardware information
- OS information
- Process information
- CGroup information

### Logs Page

The INFO logs page of the debug Web UI is at /logs under the main statestored Web UI.

This page shows the last portion of the statestored.INFO log file, including the info, warning, and error logs for the statestored. You can refer here to see the details of the most recent operations, whether the operations succeeded or encountered errors. This page provides one central place for the log files and saves you from looking around the filesystem for the log files, which could be in different locations on clusters that use cluster management software.

### Memz Page

The Memory Usage page of the debug Web UI is at /memz under the main statestored Web UI.

This page displays summary and detailed information about memory usage by the statestored. You can see the memory limit in effect for the node, and how much of that memory Impala is currently using.

### Metrics Page

The Metrics page of the debug Web UI is at /metrics under the main statestored Web UI.

This page displays the current set of metrics: counters and flags representing various aspects of stat estored internal operation.

### RPC Services Page

The RPC durations page of the statestored debug Web UI is at /rpcz under the main statestored Web UI.

This page displays information, such as the durations, about the RPC communications of this stat estored with other Impala daemons.

### Subscribers Page

The Subscribers page of the debug Web UI is at /subscribers under the main statestored Web UI.

This page displays information about the other Impala daemons that have registered with the stat estored to receive and send updates.

### Threadz Page

The Threads page of the debug Web UI is at /threadz under the main statestored Web UI.

This page displays information about the threads used by this instance of statestored, and shows which categories they are grouped into. Making use of this information requires substantial knowledge about Impala internals.

### Topics Page

The Topics page of the debug Web UI is at /topics under the main statestored Web UI.

This page displays information about the topics to which the other Impala daemons have registered to receive updates.

### Varz Page

The Varz page of the debug Web UI is at /varz under the main statestored Web UI.

This page shows the configuration settings in effect when this instance of statestored communicates with other Hadoop components such as HDFS and YARN. These settings are collected from a set of configuration files.

The bottom of this page also lists all the command-line settings in effect for this instance of statesto red.

### Prometheus Metrics Page

At /metrics_prometheus, under the main statestored Web UI, the metrics are generated in Prometheus exposition format that Prometheus can consume for event monitoring and alerting.

The /metrics_prometheus is not shown in the Web UI list of pages.

## Debug Web UI for Catalog Server

You can use the Catalog Server (`catalogd`) Web UI to view information about the databases, tables, and other objects managed by Impala, in addition to the resource usage and configuration settings of the catalogd.

The main page of the debug Web UI is at http://*IMPALA#SERVER#HOSTNAME*:25020/ (non-secure cluster) or https://*IMPALA#SERVER#HOSTNAME*:25020/ (secure cluster).

### Main Page

The main catalogd Web UI page at / lists the following information about the catalogd:

- The version of the catalogd daemon
- Process start time
- Hardware information
- OS information
- Process information
- CGroup information

### Catalog Page

The Catalog page of the debug Web UI is at /catalog under the main catalogd Web UI.

This page displays a list of databases and associated tables recognized by this instance of catalogd. You can use this page to locate which database a table is in, check the exact spelling of a database or table name, look for identical table names in multiple databases. The catalog information is represented as the underlying Thrift data structures.

### JMX

The JMX page of the catalogd debug Web UI is at /jmx under the main catalogd Web UI.

This page displays monitoring information about various JVM subsystems, such as memory pools, thread management, runtime. etc.

### Java Log Level

The Change log level page of the catalogd debug Web UI is at /log_level under the main catalogd Web UI.

The page displays the current Java and backend log levels and allows you to change the log levels dynamically without having to restart the catalogd

### Logs Page

The INFO logs page of the debug Web UI is at /logs under the main catalogd Web UI.

This page shows the last portion of the catalogd.INFO log file, including the info, warning, and error logs for the catalogd daemon. You can refer here to see the details of the most recent operations, whether the operations succeeded or encountered errors. This page provides one central place for the log files and saves you from looking around the filesystem for the log files, which could be in different locations on clusters that use cluster management software.

### Memz Page

The Memory Usage page of the debug Web UI is at /memz under the main catalogd Web UI.

This page displays summary and detailed information about memory usage by the catalogd. You can see the memory limit in effect for the node, and how much of that memory Impala is currently using.

### Metrics Page

The Metrics page of the debug Web UI is at /metrics under the main catalogd Web UI.

This page displays the current set of metrics: counters and flags representing various aspects of cata logd internal operation.

### RPC Services Page

The RPC durations page of the catalogd debug Web UI is at /rpcz under the main catalogd Web UI.

This page displays information, such as the durations, about the RPC communications of this cata logd with other Impala daemons.

### Threadz Page

The Threads page of the debug Web UI is at /threadz under the main catalogd Web UI.

This page displays information about the threads used by this instance of catalogd, and shows which categories they are grouped into. Making use of this information requires substantial knowledge about Impala internals.

### Varz Page

The Varz page of the debug Web UI is at /varz under the main catalogd Web UI.

This page shows the configuration settings in effect when this instance of catalogd communicates with other Hadoop components such as HDFS and YARN. These settings are collected from a set of configuration files.

The bottom of this page also lists all the command-line settings in effect for this instance of cata logd.

### Prometheus Metrics Page

At /metrics_prometheus, under the main catalogd Web UI, the metrics are generated in Prometheus exposition format that Prometheus can consume for event monitoring and alerting.

The /metrics_prometheus is not shown in the Web UI list of pages.

# Configuring Impala Web UI

As an administrator, you can diagnose issues with each daemon on a particular host, or perform other administrative actions such as cancelling a running query from the built-in web server's UI. The built-in web server is inlcuded within each of the Impala-related daemons. By default, these web servers are enabled. You can turn them off in a high-security configuration where it is not appropriate for users to have access to this kind of monitoring information through a web interface.

## Enabling and Disabling Access to Impala Web Servers

By default, these web servers are enabled. You might turn them off in a high-security configuration where it is not appropriate for users to have access to this kind of monitoring information through a web interface.

To enable or disable Impala Web Servers for Web UI in Cloudera Manager:

- Impala Daemon

  1. Navigate to ClustersImpala ServiceConfiguration.
  2. Select ScopeImpala Daemon .
  3. Select CategoryPorts and Addresses.
  4. Select or clear Enable Impala Daemon Web Server.
  5. Click Save Changes, and restart the Impala service.

- Impala StateStore

  1. Navigate to ClustersImpala ServiceConfiguration.
  2. Select ScopeImpala StateStore.
  3. Select CategoryMain.
  4. Select or clear Enable StateStore Web Server.
  5. Click Save Changes, and restart the Impala service.
- Impala Catalog Server

  1. Navigate to ClustersImpala ServiceConfiguration.
  2. Select ScopeImpala Catalog Server.
  3. Select CategoryMain.
  4. Check or clear Enable Catalog Server Web Server.
  5. Click Save Changes, and restart the Impala service.

## Configuring Secure Access for Impala Web Servers

Cloudera Manager supports two methods of authentication for secure access to the Impala Catalog Server, Impala Daemon, and StateStore web servers: password-based authentication and SPNEGO authentication.

Authentication for the three types of daemons can be configured independently.

## Configuring Password Authentication

1. Navigate to ClustersImpala ServiceConfiguration.
2. Search for "password" using the Search box in the Configuration tab. This should display the password-related properties (Username and Password properties) for the Impala Daemon, StateStore, and Catalog Server. If there are multiple role groups configured for Impala Daemon instances, the search should display all of them.
3. Enter a username and password into these fields.
4. Click Save Changes, and restart the Impala service.

Now when you access the Web UI for the Impala Daemon, StateStore, or Catalog Server, you are asked to log in before access is granted.

## Enabling Kerberos HTTP SPNEGO Authentication for Web UI

To provide security through Kerberos, Impala Web UIs support SPNEGO. SPNEGO is a protocol for securing HTTP requests with Kerberos by passing negotiation tokens through HTTP headers.

To enable authorization using SPNEGO in Cloudera Manager:

1. Navigate to ClustersImpala ServiceConfiguration.
2. Select ScopeImpala 1 (Service-Wid).
3. Select the Enable Kerberos Authentication for HTTP Web-Consoles field.

   This setting is effective only Kerberos is enabled for the HDFS service.
4. Click Save Changes, and restart the Impala service.

## Configuring TLS/SSL for Web UI

1. Create or obtain an TLS/SSL certificate.
2. Place the certificate, in .pem format, on the hosts where the Impala Catalog Server and StateStore are running, and on each host where an Impala Daemon is running. It can be placed in any location (path) you choose. If all the Impala Daemons are members of the same role group, then the .pem file must have the same path on every host.
3. Navigate to ClustersImpala ServiceConfiguration.
4. Search for "certificate" using the Search box in the Configuration tab. This should display the certificate file location properties for the Impala Catalog Server, Impala Daemon, and StateStore. If there are multiple role groups configured for Impala Daemon instances, the search should display all of them.

5. In the property fields, enter the full path name to the certificate file, the private key file path, and the password for the private key file..
6. Click Save Changes, and restart the Impala service.

**Important:** If Cloudera Manager cannot find the .pem file on the host for a specific role instance, that role will fail to start.

When you access the Web UI for the Impala Catalog Server, Impala Daemon, and StateStore, https will be used.

### Opening Impala Web UIs

### Procedure

1. Navigate to ClustersImpala ServiceConfiguration.
2. Open the appropriate Web UI:Select Web UIImpala Catalog Web UI.
   - To open StateStore Web UI, select Web UIImpala StateStore Web UI.
   - To open Catalog Server Web UI, select Web UIImpala Catalog Web UI.
   - To open Impala Daemon Web UI:
     a. Click the Instances tab.
     b. Click an Impala Daemon instance.
     c. Click Impala Daemon Web UI.

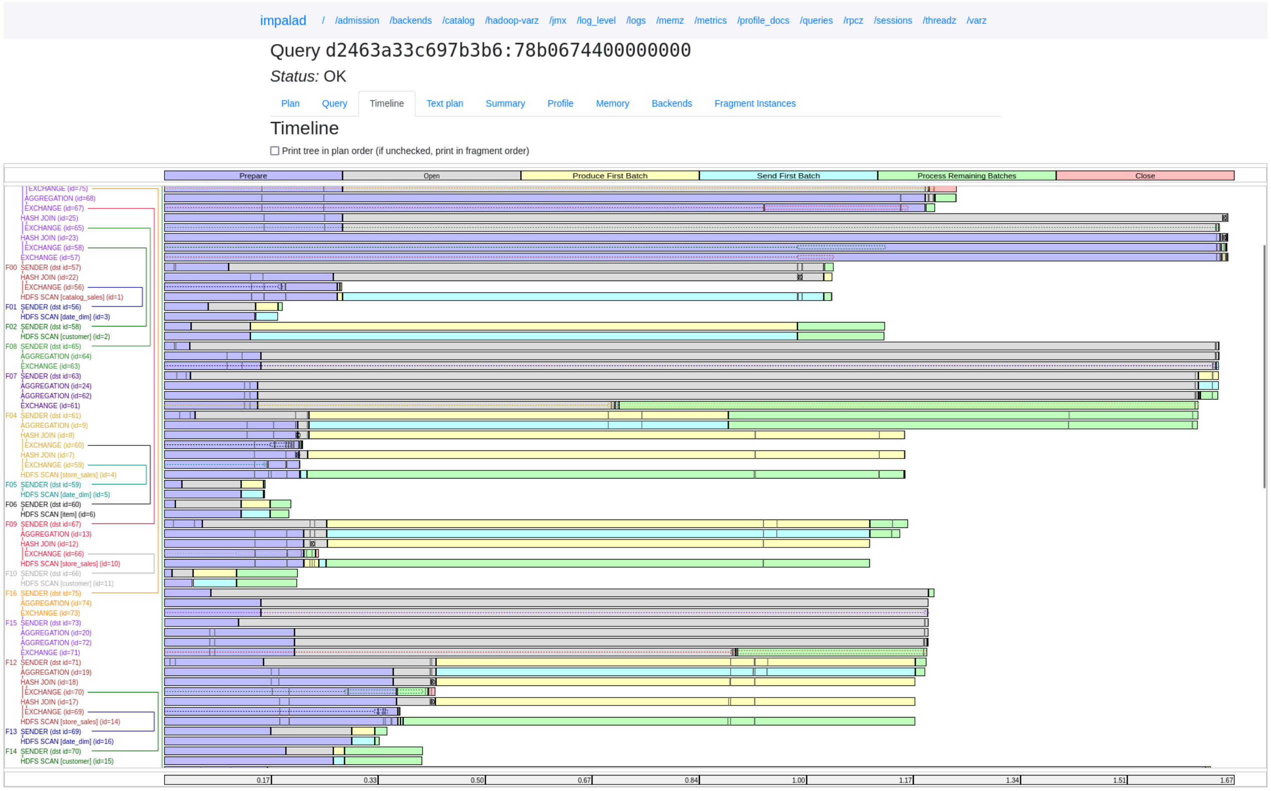
## Debug Web UI for Query Timeline

For a detailed report on how a query was executed and to understand the detailed performance characteristics of a query, you can use the built-in web server’s UI and look at the Gantt chart.

The Gantt chart is an alternative to the profile command and is a graphical display in the WebUI that renders timing information and dependencies. This chart allows you to determine where queries are spending time instead of looking for timing information in the profile. Once you identify which fragments or operators are consuming the most time you may need to look at the summary profile or full profile to drill down further.

In subsequent releases, the timeline will be enhanced to display additional details so that looking at the profiles will not be necessary.

Here is the screenshot of the Gantt chart that shows the Query timeline of a TPC_DS query.



impalad / /admission /backends /catalog /hadoop-varz /jmx /log_level /logs /memz /metrics /profile_docs /queries /rpcz /sessions /threadz /varz

Query d2463a33c697b3b6:78b0674400000000

*Status:* OK

Plan Query Timeline Text plan Summary Profile Memory Backends Fragment Instances

Timeline

□ Print tree in plan order (if unchecked, print in fragment order)

# Improvement in Catalog Observability

Cloudera on cloud7.3.1.500 brings significant enhancements to the Impala Catalog Web UI, addressing performance issues related to delays in processing Hive Metastore (HMS) events, which can lead to queries using outdated metadata

These improvements specifically target slow Data Definition Language (DDL) and Data Manipulation Language (DML) operations such as INSERT, LOAD DATA statements, and REFRESH operations. Additionally, it addresses operations stuck in various stages such as startup, among other scenarios. Prior to this release, troubleshooting these issues required analyzing logs, but now the process has been streamlined by enhancing the observability of the catalog across multiple dimensions, including the catalogd WebUI, DDL/DML profiling, and additional log reports on sluggish operations.

Key enhancements include:

- Catalogd WebUI: Utilize the catalogd WebUI to monitor Catalog operations.
- DDL/DML Profiling: Monitor DDL/DML profile for detailed execution insights within catalogd.
- Log Reports: Analyze logs for a comprehensive review of slow operations, offering valuable insights into the root causes of performance issues.

These improvements empower users with a more user-friendly and effective means of diagnosing and resolving catalog-related performance challenges.

# Impala workload management (Preview)

Learn how to enable Impala query logging in Cloudera Runtime to track queries, analyze performance, and retain execution data for better insights.

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

Cloudera Runtime provides you the option to enable logging Impala queries on an existing Virtual Warehouse or while creating a new Impala Virtual Warehouse. By logging the Impala queries in Cloudera Runtime, you gain increased observability of the workloads running on Impala, which you can use to improve the performance of your Impala Virtual Warehouses.

This feature represents a significant enhancement to query profiling capabilities. You can have Impala archive crucial data from each query's profile into dedicated database tables known as the query history table and live query table. These tables are part of the sys database and are designed to store valuable information that can later be queried using any Impala client, providing a consolidated view of both actively running and previously executed queries.

The query history table, sys.impala_query_log proves particularly useful when dissecting workloads for in-depth analysis of query performance. Unlike the limitations associated with query profiles, which are only available to the client that initiated the query, the query history table offers a comprehensive solution for querying completed queries without the need to parse the text of each query profile. Additionally, the query history table provides a comprehensive view across all Impala coordinators.

The Impala query information is stored indefinitely in the sys.impala_query_log table whereas the sys.impala_query_live table reflects the in-memory state of all Impala coordinators. Actively running and recently completed queries are stored in this table. Data is removed from this table once the query finishes and is persisted in the sys.impala_query_log table or if the coordinator is restarted. Therefore, there is a possibility that some of the records could momentarily be duplicated in both these tables.

Since the sys.impala_query_live table is stored only in-memory, recently completed queries that are not yet persisted to the sys.impala_query_log table are lost if the coordinator crashes. However, if the coordinator is shut down gracefully, then the recently completed queries are stored in the sys.impala_query_log table and are not lost.

The <onlyCoordinators> element in Impala’s Admission Control restricts a request pool to coordinators only, excluding executors. This is mainly used for querying the sys.impala_query_live table. However, these pools can still run any query, potentially exhausting coordinator resources. Proper naming is important to avoid unintended query routing. For more information, see *Apache Impala: onlyCoordinators*.

## Impala workload management table format

Learn about the available columns in the query history and live query system tables.

### Table Format

The following columns are available as part of the query history and live query system tables:

| Column Name | Description | Data Type | Sample Value |
|---|---|---|---|
| cluster_id | String specified through the Impala startup flag to uniquely identify an instance. | string | cluster-123 |
| query_id | Impala assigned query identifier. | string | 214d08bef0831e7a:3c65392400000000 |
| session_id | Impala assigned session identifier. | string | ea4f661af43993d8:587839553a41adb8 |
| session_type | Client session type. | string | HIVESERVER2 |
| hiveserver2_protocol_version | Version of the HiveServer (HS2) protocol that was used by the client when connecting. | string | HIVE_CLI_SERVICE_PROTOCOL_V6 |
| db_user | Effective user on the cluster. | string | csso_name |
| db_user_connection | Username from an authenticated client. | string | csso_name |

| Column Name | Description | Data Type | Sample Value |
|---|---|---|---|
| db_name | Name of the database being queried. | string | default |
| impala_coordinator | Name of the coordinator for the query. | string | coord-22899:27000 |
| query_status | Status of the query when it completes. | string | OK |
| query_state | Final state of the query. | string | FINISHED |
| impala_query_end_state | Final Impala state of the query. | string | FINISHED |
| query_type | Type of the query. | string | QUERY |
| network_address | Client IP address and port. | string | 127.0.0.1:40120 |
| start_time_utc | Time when the query started. Time zone is in UTC. | timestamp | 2024-07-17 17:13:46.414316000 |
| total_time_ms | Difference between the query end time and start time, in milliseconds (digits after the decimal point represent milliseconds). | decimal(18,3) | 136.121 |
| query_opts_config | List of query options stored as a single string containing comma-separated values of key-value pairs. | string | TIMEZONE=America/Los_Angeles,CLIENT_IDENTIFIER=Impala Shell v4.4.0a1 (04bdb4d) built on Mon Nov 20 10:49:35 PST 2023 |
| resource_pool | Name of the resource pool for the query. | string | default-pool |
| per_host_mem_estimate | Size, in bytes of the per-host memory estimate. | bigint | 5 |
| dedicated_coord_mem_estimate | Size, in bytes of the dedicated coordinator memory estimate. | bigint | 4 |
| per_host_fragment_instances | Comma-separated string listing each host and its fragment instances. | string | myhost-1:27000=1,myhost-2:27001=2 |
| backends_count | Count of the number of backends used by this query. | integer | 2 |
| admission_result | Result of the admission (not applicable to DDLs). | string | Admitted immediately |
| cluster_memory_admitted | Cluster memory, in bytes that was admitted. | integer | 4 |
| executor_group | Name of the executor group. | string | executor_group |
| executor_groups | List of all executor groups including the groups that were considered and rejected as part of Workload Aware Auto Scaling. | string | executor_group1, executor_group2… |
| exec_summary | Full text of the executor summary. | string | |
| num_rows_fetched | Number of rows fetched by the query. | bigint | 6001215 |
| row_materialization_rows_per_sec | Count of the number of rows materialized per second. | bigint | 3780 |
| row_materialization_time_ms | Time spent materializing rows converted to milliseconds. | decimal(18,3) | 1.58 |
| compressed_bytes_spilled | Count of bytes that were written (or spilled) to scratch disk space. | bigint | 241515 |

| Column Name | Description | Data Type | Sample Value |
|---|---|---|---|
| event_planning_finished | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 27.253 |
| event_submit_for_admission | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 30.204 |
| event_completed_admission | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 30.986 |
| event_all_backends_started | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 31.969 |
| event_rows_available | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 31.969 |
| event_first_row_fetched | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 135.175 |
| event_last_row_fetched | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 135.181 |
| event_unregister_query | Event from the timeline.The value represents the number of milliseconds since the query was received. | decimal(18,3) | 141.435 |
| read_io_wait_total_ms | Total read I/O wait time converted to milliseconds. | bigint | 15.091 |
| read_io_wait_mean_ms | Average read I/O wait time across executors converted to milliseconds | bigint | 35.515 |
| bytes_read_cache_total | Total bytes read from the data cache | bigint | 45823 |
| bytes_read_total | Total bytes read | bigint | 745227 |
| pernode_peak_mem_min | Minimum value of all the per-node peak memory usages | bigint | 5552846 |
| pernode_peak_mem_max | Maximum value of all the per-node peak memory usages | bigint | 5552846 |
| pernode_peak_mem_mean | Mean value of all the per-node peak memory usages | bigint | 5552846 |
| sql | SQL statement as provided by the user | string | SELECT db_user, total_time_ms from impala_query_log where query_state = 'EXCEPTION'; |
| plan | Full text of the query plan | string | |
| tables_queried | Comma-separated string containing all the tables queried in the SQL statement. Aliased tables are resolved to their actual table names. | string | db.tbl,db.tbl |

| Column Name | Description | Data Type | Sample Value |
|---|---|---|---|
| select_columns | Comma-separated string containing all columns from the select list of the sql. Aliased columns are resolved to their actual column names. Each column is in the format database.table.column_name. | string | db.tbl.col1,db.tbl.col2 |
| where_columns | Comma-separated string containing all columns from the where list of the sql. Aliased columns are resolved to their actual column names. Each column is in the format database.table.column_name. | string | db.tbl.col1,db.tbl.col2 |
| join_columns | Comma-separated string containing all columns from the sql used in a join. Aliased columns are resolved to their actual column names. Each column is in the format database.table.column_name. | string | db.tbl.col1,db.tbl.col2 |
| aggregate_columns | Comma-separated string containing all columns from the group by and having lists of the sql. Aliased columns are resolved to their actual column names. Each column is in the format database.table.column_name. | string | db.tbl.col1,db.tbl.col2 |
| orderby_columns | Comma-separated string containing all columns from the order by list of the sql. Aliased columns are resolved to their actual column names. Each column is in the format database.table.column_name. | string | db.tbl.col1,db.tbl.col2 |
| coordinator_slots | Number of query slots used by the query on the coordinator. | bigint | 1 |
| executor_slots | Numberof query slots used by the query on the executors. The value in this column represents the slots used by a single executor, not the total number of slots across all executors. | bigint | 1 |

## Impala workload management table maintenance

Understand the maintenance requirements for the sys.impala_query_live and sys.impala_query_log tables.

For efficient query performance, different maintenance needs apply to the sys.impala_query_live and sys.impala_query_log tables.

- Sys.impala_query_live:
  - No maintenance is required because it resides entirely in memory.

- Sys.impala_query_log:
  - As an Iceberg table, it requires periodic maintenance, such as:
    - Computing statistics.
    - Optimizing the table structure.
    - Performing snapshot expiration or cleanup.

Since Impala workloads are unique, no automatic maintenance is performed on the sys.impala_query_log table. You should schedule maintenance tasks according to your workload needs.

To optimize the Impala query log, run the query OPTIMIZE TABLE sys.impala_query_log (FILE_SIZE_THRESHOLD_MB=128). Cloudera recommends testing this query in the development or test environments to evaluate its impact on your workload. For best results, run the query during low cluster activity times.

## Impala workload management use cases

Learn how to use query history to track executed queries by user, identify frequently queried statements, and report long-running queries for analysis.

A consolidated view of reports from previously executed queries can be useful in the following use cases:

- Collecting history of all queries run and reporting by user, start/end time, and execution duration. For instance:

```
SELECT db_user, start_time, end_time, total_time_ms, sql FROM impala_que
ry_log ORDER BY db_user;
```

- Collecting the top five most frequently executed queries. For instance:

```
SELECT lower(sql) sql, count(sql) count FROM sys.impala_query_log GROUP BY
 lower(sql) ORDER BY count desc LIMIT 5;
```

- Reporting on queries that ran over 10 minutes. For instance:

```
SELECT db_user, total_time_ms, sql FROM sys.impala_query_log WHERE total
_time_ms > 600000;
```

- Reporting on queries that are actively running and have been running for over 10 minutes. For instance:

```
SELECT db_user, total_time_ms, sql FROM sys.impala_query_live WHERE tota
l_time_ms > 600000;
```

## Enable Impala workload management

Learn how to enable and configure Impala query logging to store and analyze query history.

To use this feature, enable Impala query logging while creating a new Virtual Warehouse or by editing an existing one by selecting the Log Impala Queries option. By default, the Log Impala Queries option is off.

You can then configure the Impala coordinator using specific startup flags to store query history. Impala manages the table serving as a centralized repository for all query histories across databases. Completed queries are periodically inserted into this table based on a preconfigured interval.

This feature streamlines the process of query history management, providing a more accessible and comprehensive way to analyze and retrieve information about completed queries.

**Note:** This feature is available from versions 2024.0.18.0 (tech preview), 2025.0.19.0 and higher (GA).

**Important:** The following query types are not written into the query logging tables:

- SET
- SHOW
- USE
- DESCRIBE

## Impala coordinator Startup flags

Learn about Impala coordinator startup flags used to create and configure query logging tables during startup.

On startup, each Impala coordinator runs an SQL statement to create the query logging tables. The following table lists the Impala startup coordinator flags that you can configure:

| Name | Data Type | Default | Description |
|---|---|---|---|
| cluster_id | string | | Specifies an identifier string that uniquely represents this cluster. This identifier is included in both the tables and is used as a table partition for the sys.impala_query_log table. |
| query_log_shutdown_timeout_s | number (seconds) | 30 | Hidden flag. Number of seconds to wait for the queue of completed queries to be carried into the query history table before timing out and continuing the shutdown process.The query history table drain process runs after the shutdown process completes, therefore the max shutdown time is extended by the value specified in this flag. |
| workload_mgmt_user | string | impala | Specifies the user that will be used to insert records into the query history table. |
| query_log_write_interval_s | number (seconds) | 300 | Number of seconds to wait before inserting completed queries into the query history table. Allows for batching inserts to help avoid small files. |
| query_log_max_queued | number | 3000 | Maximum number of completed queries that can be queued before they are written to the query history table.This flag operates independently of the query_log_write_interval_m flag. If the number of queued records reaches this value, the records will be written to the query log table no matter how much time has passed since the last write.A value of 0 indicates no maximum number of queued records. |

| Name | Data Type | Default | Description |
|---|---|---|---|
| query_log_max_sql_length | number | 16777216 | Maximum length of a SQL statement that will be recorded in the completed queries table. If a SQL statement with a length longer than this specified value is executed, the SQL inserted into the completed queries table will be trimmed to this length. Any characters that require escaping will have their backslash character counted towards this limit. |
| query_log_max_plan_length | number | 16777216 | Maximum length of the SQL query plan that will be recorded in the completed queries table. If a query plan has a length longer than this value, the plan inserted into the completed queries table will be trimmed to this length. Any characters that require escaping will have their backslash character counted towards this limit. |
| query_log_request_pool | string | | Specifies a pool or queue used by the queries that insert into the query log table. Empty value causes no pool to be set. |
| query_log_dml_exec_timeout_s | number | 120 | Specifies the value of the EXEC_TIME_LIMIT_S query option on the DML that inserts records into the sys.impala_query_log table. |