Job Retry (Preview)

Date published: 2025-08-25
Date modified: 2025-09-04

Legal Notice

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.

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

Using Job Retry

The Job Retry feature enables automatic retries of jobs based on their terminal execution states, namely failed, timed out, or skipped. It also supports concurrent execution of job retry runs, ensuring that scheduled job runs remain unaffected and are not blocked by retry processes. Users have the flexibility to configure various options to define the retry behavior. These retries are fully automated, eliminating the need for manual intervention.

Enabling Job Retry Entitlement

This feature is behind entitlement. Use **ML_ENABLE_JOB_RETRY** to enable this feature.

Configuring Job Retry settings

Retry job runs are designed to operate asynchronously, ensuring they do not disrupt the normal flow of a job run. These retries are executed concurrently to maintain efficiency.

Administrators can define default values for the Job Retry parameters, and only the Administrator can configure a hard limit on the maximum number of retry job runs that can be executed alongside normal job runs. This setting must only be enabled if you want to manage and limit resource usage for retry job runs.

Steps

1. In the **Cloudera** console, click the **Cloudera AI** tile.
   The **Cloudera AI Workbenches** page displays.
2. Click on the name of the workbench.
   The workbench **Home** page displays.
3. Select **Site Administration** in the left Navigation pane.
4. Select the **Settings** tab.
5. Select **Job Retry Configuration → Limit Concurrent Retries.**
6. Enable ***Limit Concurrent Retries*** by selecting the checkbox.

   Enabling this option sets a maximum limit value to the number of retry job runs that can be active at the same time.
7. Define the limit value for the **Maximum *Concurrent Retry Limit*** option.

   The **Maximum Concurrent Retry Limit** specifies the maximum number of retry job runs that can execute concurrently across the entire workbench, regardless of the total number of jobs running.

If the maximum limit value defined as **Maximum *Concurrent Retry Limit*** is reached, any additional retry job runs are rescheduled until the number of active retry runs falls below the limit.
Enable a hard limit only if retry job runs are consuming excessive resources; otherwise, avoid setting a hard limit.
Administrators can set the maximum limit value if the **Limit Concurrent Retires** option is enabled.

8. Configure the **Default Settings for all jobs** section**.**

   If the administrator configures these settings, the specified values automatically populate the fields in the new job creation form when a user creates a job. In this case, the Job Retry settings act as default values that the administrator can recommend to users.

   If the administrator does not configure these settings, the fields remain blank in the new job creation form.

   In both cases, users have the flexibility to customize these values during job creation or update them later through the job settings page.

9. Select ***Enable Retry*** to enable a retry run for the job.
   Define the following parameters for Job Retry:
   - **Maximum Retry –** It is the maximum number of retry attempts that can be triggered for a single job run in case of continuous failure of retry job runs.
     - The m*inimum value is 1.*
   - **Retry Delay (minutes) –** It is the delay between two consecutive retry job runs for a failed instance of the run.
     - The m*inimum value is 1 minute.*
   - **Retry Conditions:** It controls the terminal states of a job run that trigger a retry. The Retry process completes as soon as any of the selected criteria is met. Select at least one of the following criteria if **Retry** is enabled, but you can select any combination of the following **Retry Conditions** options:
     i. Script Failure – It runs the Retry process for user script failures if the user script exits with a non-zero exit code after the execution of the script.
     ii. System Failure – It runs the Retry process for any kind of system- or engine-related failures not including user script failures.
     iii. Timed-out Runs – It runs the Retry process for timed-out job runs.
     iv. Skipped Runs – It runs the Retry process for skipped job runs.
10. Click the ***Update*** *button* to save the settings*.*

Related links

- Creating a Job
- Job Retry Parameters

Job Retry parameters

If the Administrator configures the Job Retry settings, the specified values automatically populate the fields in the new job creation form when a user creates a job. In this case, the Job Retry settings act as default values that the administrator can recommend to users.

If the administrator does not configure the Job Retry settings, the fields remain blank in the new job creation form.

In both cases, users have the flexibility to customize the values during job creation or update them later through the job settings page.

Parameter	**Description**
Maximum Retry	The maximum number of retries that can be performed for a single failed job run. Retries continue until either a job run succeeds or the total number of retries reaches the maximum retry count specified by the user. Setting the Maximum Retry option to a high value can result in higher resource usage. The value must always be greater than 0.
Retry Delay	The Retry Delay value defines the time between each subsequent retry attempt in minutes. The Retry Delay period ensures that the transient errors (for example, temporary network or resource outage) get fixed without overloading the system with job run requests. If you encounter transient issues, set the Retry Delay parameter to a higher value.

	If you address script failures and transient issues are not a concern, a lower value for the retry delay can be configured. If you have time-sensitive Jobs, set the Retry Delay parameter to a smaller value to trigger the retry at a faster pace. The value must always be greater than 0, and the minimum retry delay value is 1 minute.
Retry Conditions	Retry Conditions configure this parameter to control the terminal states of a job run that trigger a retry. Select minimum one of the criteria if **Retry** is enabled, but you can select any combination of the **Retry Conditions** options. The Retry process completes as soon as any of the selected criteria is met. The following Retry Conditions options can be enabled: ○ Script Failure – It runs the Retry process for user script failures if the user script exits with a non-zero exit code after the execution of the script. ○ System Failure – It runs the Retry process for any kind of system- or engine-related failures, not including user script failures. ○ Timed-out Runs – It runs the Retry process for timed-out job runs. The timeout value must be set

	to a reasonable duration. If the value is too short, each retry will encounter the same limit, potentially resulting in a continuous timeout → retry → timeout loop. ○ Skipped Runs – It runs the Retry process for skipped job runs.
Limit Concurrent Retries	This parameter can only be set by the Administrator. It defines the limit value for the ***Max Concurrent Retry Limit parameter.*** The **Maximum Concurrent Retry Limit** parameter specifies the maximum number of retry job runs that can execute concurrently across the entire workbench, regardless of the total number of jobs running. If the maximum limit value defined as ***Max Concurrent Retry Limit*** is reached, any additional retry job runs are rescheduled until the number of active retry runs falls below the limit. Enable a hard limit only if retry job runs are consuming excessive resources; otherwise, avoid setting a hard limit. Administrators can set this value if the **Limit Concurrent Retries** option is enabled.

Creating a Job

This topic describes how to automate analytics workloads with a built-in job and pipeline scheduling system that supports real-time monitoring, job history, and email alerts.

A *job* automates the action of launching an engine, running a script, and tracking the results, all in one batch process. Jobs are created within the purview of a single project and can be configured to run on a recurring schedule. You can customize the engine environment for a job, set up email alerts for successful or failed job runs, and email the output of the job to yourself or a colleague.

Jobs are created within the scope of a project. When you create a job, you are asked to select a script to run as part of the job and create a schedule for when the job is planned to run. Optionally, you can configure a job to be dependent on another existing job, thus creating a pipeline of tasks to be accomplished in a sequence.
Note: The script files and any other job dependencies must exist within the scope of the same project.

Creating a Job using UI

1. Navigate to the project for which you want to create a job.
2. On the left-hand sidebar, click **Jobs**.
3. Click the **New Job** button.
4. Enter a **Name** for the job.
5. In **Run Job as**, if the job is to run in a service account, select the **Service Account** radio button and choose the account from the dropdown menu.
6. In **Script**, select a script to run for this job by clicking on the folder icon. You are able to select a script from a list of files that are already part of the project. To upload more files to the project, see **Managing Project** Files.
7. In **Arguments**, enter arguments to provide to the script.
   Use the environment variable `JOB_ARGUMENTS` to access the arguments in a runtime-agnostic way.
   For non-PBJ Runtimes, the contents of this field are available as standard command-line arguments.
8. Depending on the code you are running, select a **Runtime Kernel** for the job from one of the following options: Python 3.
9. Select one of the following options for a **Schedule** for the job runs:
   - **Manual** - Select this option if you plan to run the job manually each time.
   - **Recurring** - Select this option if you want the job to run in a recurring pattern every X minutes, or on an hourly, daily, weekly or monthly schedule. Set the recurrence interval with the dropdown buttons.
     As an alternative, select the Use a cron expression checkbox and enter a

Unix-style cron expression to set the interval. The expression must have five fields, specifying the minutes, hours, day of month, month, and day of week. If the cron expression is deselected, the schedule indicated in the drop-down settings takes effect.

- **Dependent** - Use this option when you are building a pipeline of jobs to run in a predefined sequence. From a dropdown list of existing jobs in this project, select the job that this one depends on. Once you have configured a dependency, this job will run only after the preceding job in the pipeline has completed a successful run.

10. Configure the **Job Retry Settings** options at each job level. This option is available only if the **Schedule** option is set to **Recurring** for the job.

    - Select ***Enable Retry*** to enable a retry run for the job.
    - Configure the following Job Retry settings:
        - **Maximum Retry –** The maximum number of retry attempts that can be triggered for a single job run in case of continuous failure of retry job runs.
            - *The minimum value is 1.*
        - **Retry Delay (minutes) – The** delay between two consecutive retry job runs for a failed instance of the run.
            - The *minimum value is 1 minute.*
        - **Retry Conditions –** Different options can be configured to control the terminal states of a job run that trigger a retry. The Retry process completes as soon as at least one (or more) option is selected.
          Select at least one of the following criteria if **Retry** is enabled, but you can select any combination of the following **Retry Conditions** options:
            1. Script Failure – Runs the Retry process for user script failures if the user script exits with a non-zero exit code after the execution of the script.
            2. System Failure – Runs the Retry process for any kind of system- or engine-related failures, not including user script failures.
            3. Timed-out Runs – Runs the Retry process for timed-out job runs.
            4. Skipped Runs – Runs the Retry process for skipped job runs.

11. Select a **Resource Profile** to specify the number of cores and memory available for each session.
12. Optionally, in the **Timeout in Minutes** field, enter a timeout value in minutes.
13. Click **Set environment** variables if you want to set any values to override the overall project environment variables.

This document has been released as part of a technical preview for features described herein. Technical preview components are provided as a convenience to our customers for their evaluation and trial usage. These components are provided 'as is' without warranty or support. Further, Cloudera assumes no liability for the usage of technical preview components, which should be used by customers at their own risk.

14. Specify a list of **Job Report Recipients** to whom you can send email notifications with detailed job reports for job success, failure, or timeout. You can send these reports to yourself, your team if the project was created under a team account, or any other external email addresses.
15. Add any **Attachments,** such as the console log, to the job reports that will be emailed.
16. Click the **Create Job** button.
    You can use the API v2 to schedule jobs from third-party workflow tools. For more information, see **Using the Jobs API**.

Related information

- Managing Project Files
- Using the Jobs API
- Legacy Jobs API (Deprecated)

Creating a Job using API

Create a job using the API by following the steps included in the code.

```
job_body = cmlapi.CreateJobRequest()


# name and script

job_body.name = "my job name"

job_body.script = "pi.py"


# arguments

job_body.arguments = "arg1 arg2 \"all arg 3\""


# engine kernel

job_body.kernel = "python3" # or "r", or "scala"


# schedule

# manual by default

# for recurring/cron:
```

```
job_body.schedule = "* * * * 5" # or some valid cron string

# job retry settings

job_body.retry_enabled= true
job_body."max_retry"= 1
job_body."retry_delay"= 1
job_body.retry_for_script_failure= true
job_body.retry_for_skipped_runs= true
job_body.retry_for_system_failure= true
job_body.retry_for_timedout_runs= true

# for dependent (don't set both parent_job_id and schedule)
job_body.parent_job_id = "abcd-1234-abcd-1234"

# resource profile (cpu and memory can be floating point for partial)
job_body.cpu = 1 # one cpu vcore
job_body.memory = 1 # one GB memory
job_body.nvidia_gpu = 1 # one nvidia gpu, cannot do partial gpus

# timeout
job_body.timeout = 300 # this is in seconds

# environment
job_body.environment = {"MY_ENV_KEY": "MY_ENV_VAL", "MY_SECOND_ENV_KEY": "MY_SECOND_ENV_VAL"}
```

```
    # attachment

    job_body.attachments = ["report/1.txt", "report/2.txt"] # will
    attach /home/cdsw/report/1.txt and /home/cdsw/report/2.txt to
    emails



    # After setting the parameters above, create the job:

    client = cmlapi.default_client("host", "api key")

client.create_job(job_body, project_id="id of project to create job
in")
```

For more examples of commands related to jobs, see **Using the Jobs API**.

Related information

- Managing Project Files
- Using the Jobs API
- Legacy Jobs API (Deprecated)

Viewing Job History

You can view the history for jobs that run within a project.

1. Navigate to the project where the job was created.
   The project page displays.
2. Click **Jobs** on the Project page.
3. Select the relevant job.
4. Click the **History** tab. You see a list of all the job runs with some basic information, such as who created the job, run duration, and status. Click individual runs to see the session output for each run.

   (Optional) If **Job Retry** is enabled, click the **Retry** tag to view the failed run that triggered the retry. On the **Job History** page, job runs that were retried are marked with a **Retry** tag. This option is visible only if retries are enabled and have been triggered.