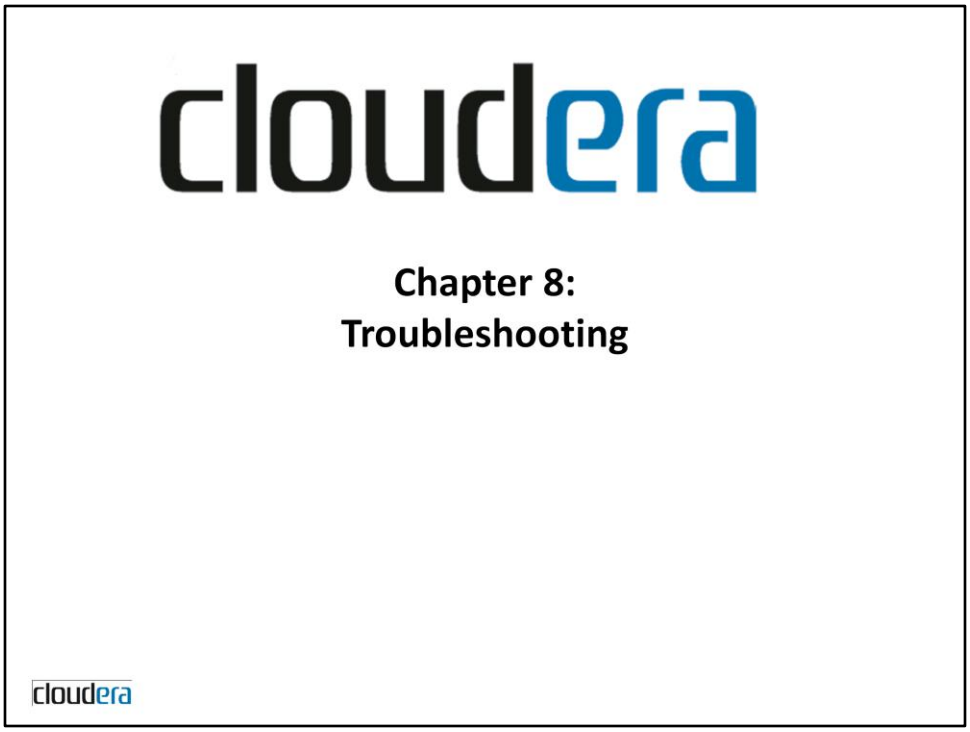


In this chapter, you will learn

- How to use the JobTracker Web interface
- How to use the logs
- Other troubleshooting tips

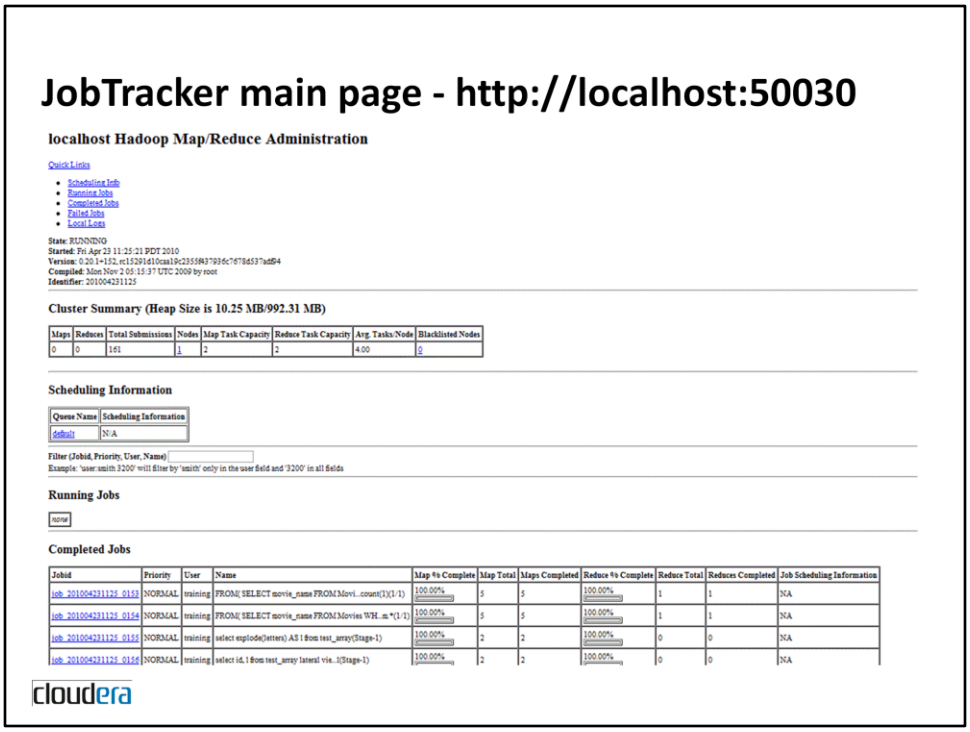
cloudera

Hadoop's JobTracker Web interface

- Hive gives minimal information when a job fails
  - `FAILED: Execution Error, return code 2 from org.apache.hadoop.hive.ql.exec.ExecDriver`
- But Hadoop has a Web interface
  - http://localhost:50030/jobtracker.jsp

cloudera

If a Hive job fails, minimal information is returned to the console. However, the JobTracker in the Hadoop cluster has a Web interface which exposes what jobs are running or have run. This is useful for troubleshooting a failed job.



The main page of the Job Tracker Web UI shows completed and running jobs. Find the job by its jobid (this is printed to the Hive console when a job starts).

Finding the logs

- Click on the job under "Failed Jobs"

Failed Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed
job_201004231125_0158	NORMAL	training	insert overwrite table test_custom_map s...t(Stage-1)	100.00%	5	0

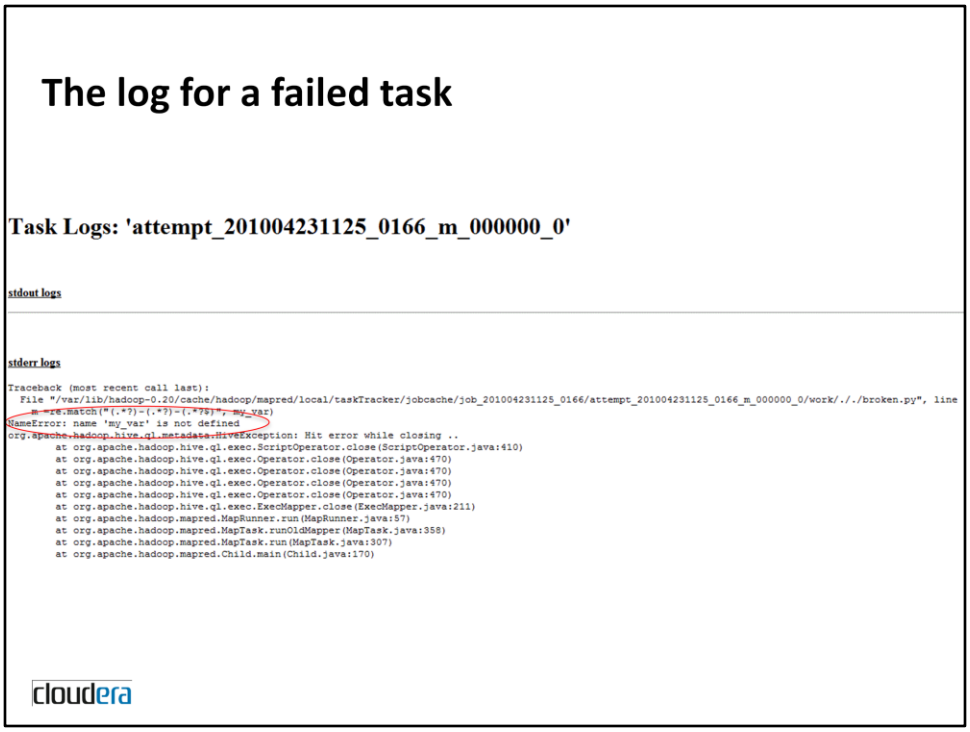
- Find the last 4KB of logs

Hadoop job_201004231125_0166 failures on localhost

Attempt	Task	Machine	State	Error	Logs
attempt_201004231125_0166_m_000000_0	task_201004231125_0166_m_000000	localhost	FAILED	java.lang.RuntimeException: Error while closing operators at org.apache.hadoop.hive.ql.exec.ExecMapper.close(ExecMapper.java:282) at org.apache.hadoop.mapred.MapRunner.run(MapRunner.java:57) at org.apache.hadoop.mapred.MapTask.runOldMapper(MapTask.java:358) at org.apache.hadoop.mapred.MapTask.run(MapTask.java:307) at org.apache.hadoop.mapred.Child.main(Child.java:170) Caused by: org.apache.hadoop.hive.ql.metadata.HiveException: Hit error while closing .. at org.apache.hadoop.hive.ql.exec.ScriptOperator.close(ScriptOperator.java:416) at org.apache.hadoop.hive.ql.exec.Operator.close(Operator.java:470) at org.apache.hadoop.hive.ql.exec.Operator.close(Operator.java:470) at org.apache.hadoop.hive.ql.exec.Operator.close(Operator.java:470) at org.apache.hadoop.hive.ql.exec.Operator.close(Operator.java:470) at org.apache.hadoop.hive.ql.exec.ExecMapper.close(ExecMapper.java:211) ... 4 more	Last 4KB Last 8KB All

cloudera

To find the relevant logs, first click on the job link. This will bring you to the job page. Then choose a task that failed. From the task page, find the logs in the right-hand column.



A task can fail for many reasons. For example, Hive may not be able to read the data in a table due to incorrect field terminators. Or you may have added (or not added) files to the distributed cache that causes Classpath issues. Or in this case, a custom map script had a typo.

Additional logs

- By default, Hive logs to `/tmp/{user.name}/hive.log`
- Information can also be sent to the console

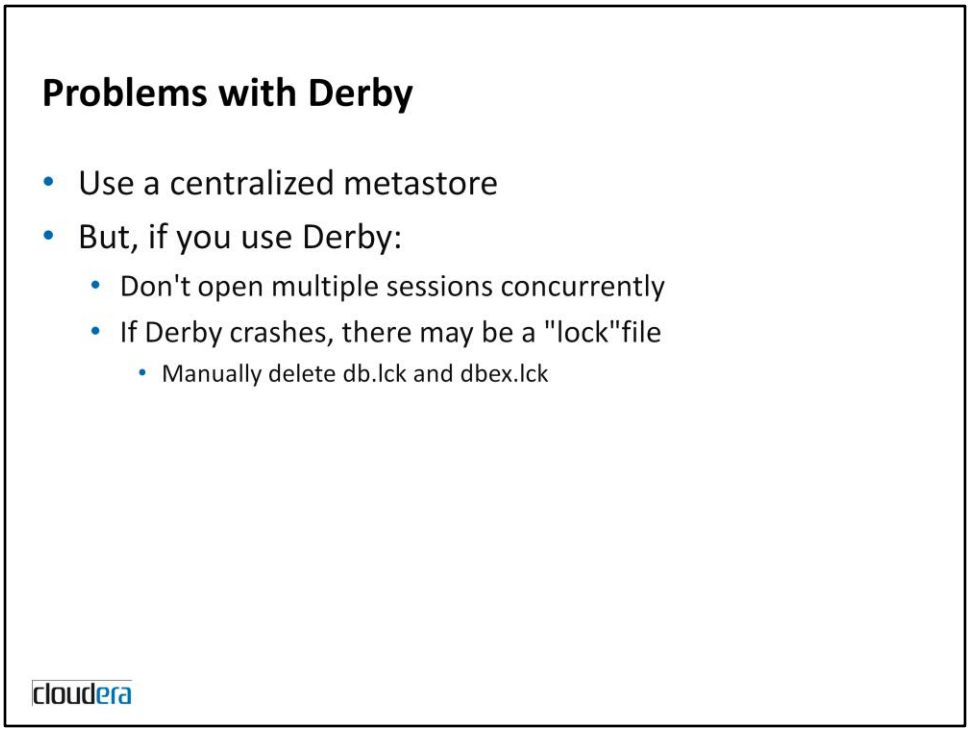
```
hive -hiveconf hive.root.logger=INFO,console
```

- Enabling logging cannot be done dynamically with a SET

cloudera

Hive creates a log file which defaults to `/tmp/{user.name}/hive.log`. Sometimes it is more useful to have verbose logging to the console. The configuration setting hive.root.logger controls the level of logging as well as the location. "`hive.root.logger=INFO,console`" means that the INFO level of logging should be used and the messages sent to the console instead of the log file.

This configuration setting cannot be enabled dynamically (it cannot be turned on via `SET`). It is necessary to logout and use the `-hiveconf` option or edit the `hive-site.xml` file.



It is highly recommended that a centralized metastore (e.g., MySQL) be configured early on. However, when using the Derby metastore, there are some common issues that can occur. First, do not attempt to use two Hive shell instances concurrently. Also, if Derby crashes, it is possible it did not have a chance to remove its "lock" file. This file is used to tell Derby that the database is in use. If this happens, just manually delete the `db.lck` and `dbex.lck` files.

Pseudo-distributed mode or LocalJobRunner

- It is common to test things in pseudo-distributed mode
  - E.g., running a Hadoop cluster locally
  - Some things could work that will fail on a real cluster
- If the JobTracker has not been configured in `mapred-site.xml`, the default is a LocalJobRunner
  - LocalJobRunner runs everything in a single JVM

cloudera

Often developers want to test their code locally on a subset of the data before deploying to the Hadoop cluster. Pseudo-distributed mode refers to a Hadoop cluster running on one machine. A LocalJobRunner means a single JVM process runs everything (there is not a separate JobTracker, TaskTracker, NameNode or DataNode). Be aware that some things could fail on the cluster even though they worked locally. For example, if you are using a custom map or reduce script and forgot to add that file to Hadoop's distributed cache via `ADD FILE`. In pseudo-distributed mode or in the LocalJobRunner, the script will be available since you are on a single machine. But in a real cluster, the cluster nodes will fail to find your script.

Use the mailing list

- http://hadoop.apache.org/hive/mailing_lists.html
- When asking for help, good to include:
  - Version of Hive
  - Type and version of metastore
  - SET -v;
  - Enable logging to the console

cloudera

The Hive mailing list can be a great place to ask questions. If you are troubleshooting a problem, it is recommended to include as much information as possible about your issue. For example:

1. Include which version of Hive (and Hadoop) you are using
2. Also include the version and type of metastore
3. A dump of all settings is useful with `SET -v;`
4. Enabling logging to the console via hive.root.logger and then reproducing the issue may give additional useful debugging information

Conclusion

In this chapter, you have learned:

- How to use the JobTracker Web interface
- How to use the logs
- Other troubleshooting tips

cloudera