
Qubole Data Service Documentation

Release 1.0

Qubole

February 15, 2016

1 Qubole Quick Start Guide	1
1.1 AWS Quick Start Guide	1
1.2 Running a Hadoop Job	6
1.3 Running a Hive Query	9
1.4 Running Spark Applications	11
1.5 Running Spark Applications in Notebooks	15
1.6 Running a Pig Job	19
1.7 Running a Shell Command	20
1.8 Running a Mahout Job	21
1.9 Running a Dumbo Job	22
1.10 Uploading a File to Amazon S3 Buckets	24
1.11 Azure Quick Start Guide	26
1.12 GCE Quick Start Guide	36
2 Qubole User Guide	45
2.1 Introduction	45
2.2 Features	45
2.3 Qubole Clusters	186
2.4 Hive in Qubole	199
2.5 Hadoop in Qubole	214
2.6 HBase in Qubole	224
2.7 Pig in Qubole	237
2.8 Presto in Qubole	238
2.9 Spark in Qubole	251
2.10 Qubole Scheduler	261
2.11 Qubole Billing Guide	265
3 Qubole Administration Guide	269
3.1 Introduction	269
3.2 Using SAML Single SignOn and Google Authorization Service	269
3.3 Managing the Account Settings	270
3.4 Managing Access Permissions and Roles	271
3.5 Managing a User Profile	272
3.6 Adding and Managing Users	273
3.7 Creating and Managing Groups	274
3.8 Enabling Encryption for Data at Rest on Amazon S3	274
3.9 Installing the Qubole ODBC Driver	275
3.10 QDS Administration How-to Topics	283

4	Qubole REST API Reference	291
4.1	Overview	291
4.2	Access URL	291
4.3	Authentication	291
4.4	API Types	291
4.5	Command API	292
4.6	Hive Metadata API	324
4.7	Cluster API	331
4.8	DbTap API	369
4.9	Scheduler API	374
4.10	Reports API	395
5	Qubole FAQs	407
5.1	General Questions	407
5.2	Questions about Hive	410
5.3	Questions about Hadoop Clusters	412
5.4	Questions about Security	414
	HTTP Routing Table	417

Qubole Quick Start Guide

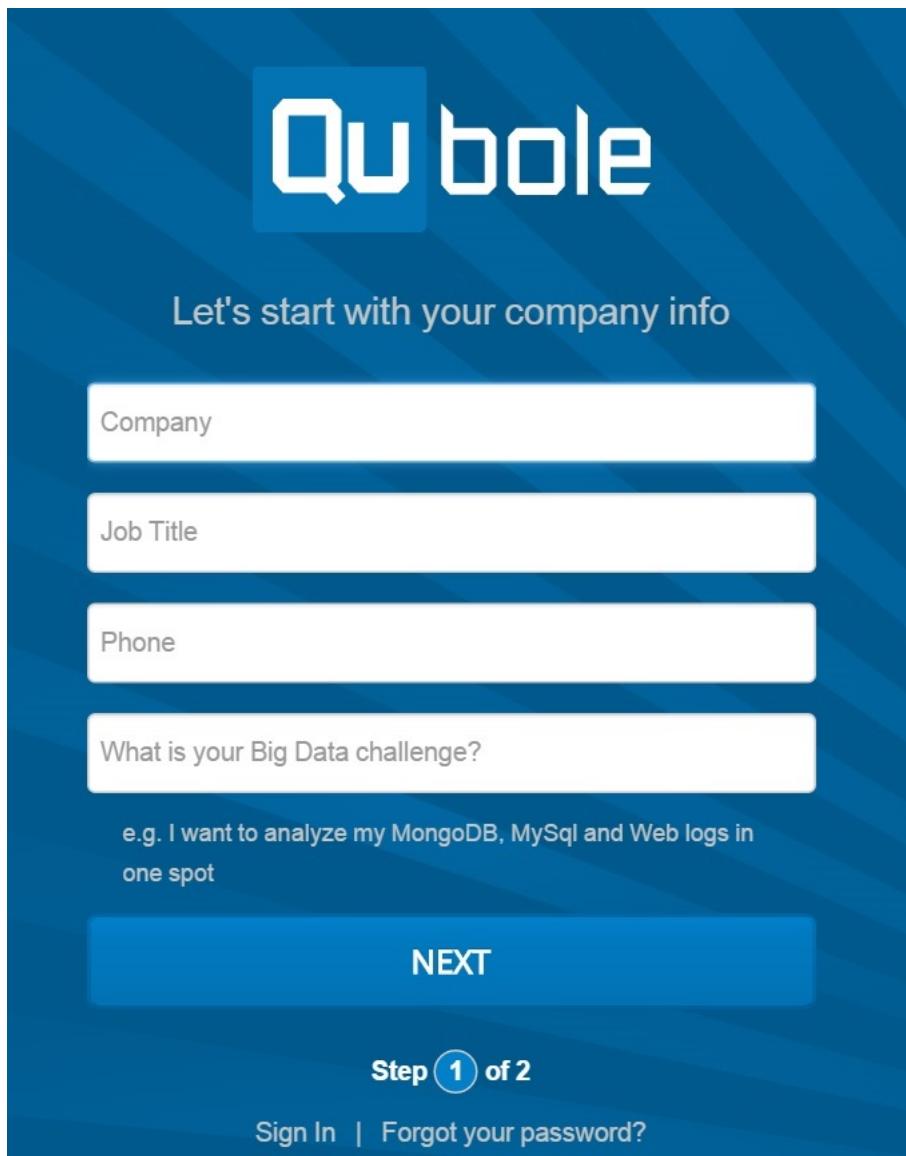
1.1 AWS Quick Start Guide

This document is intended for new users to quickly start up on Qubole Data Services by running simple Hive queries.

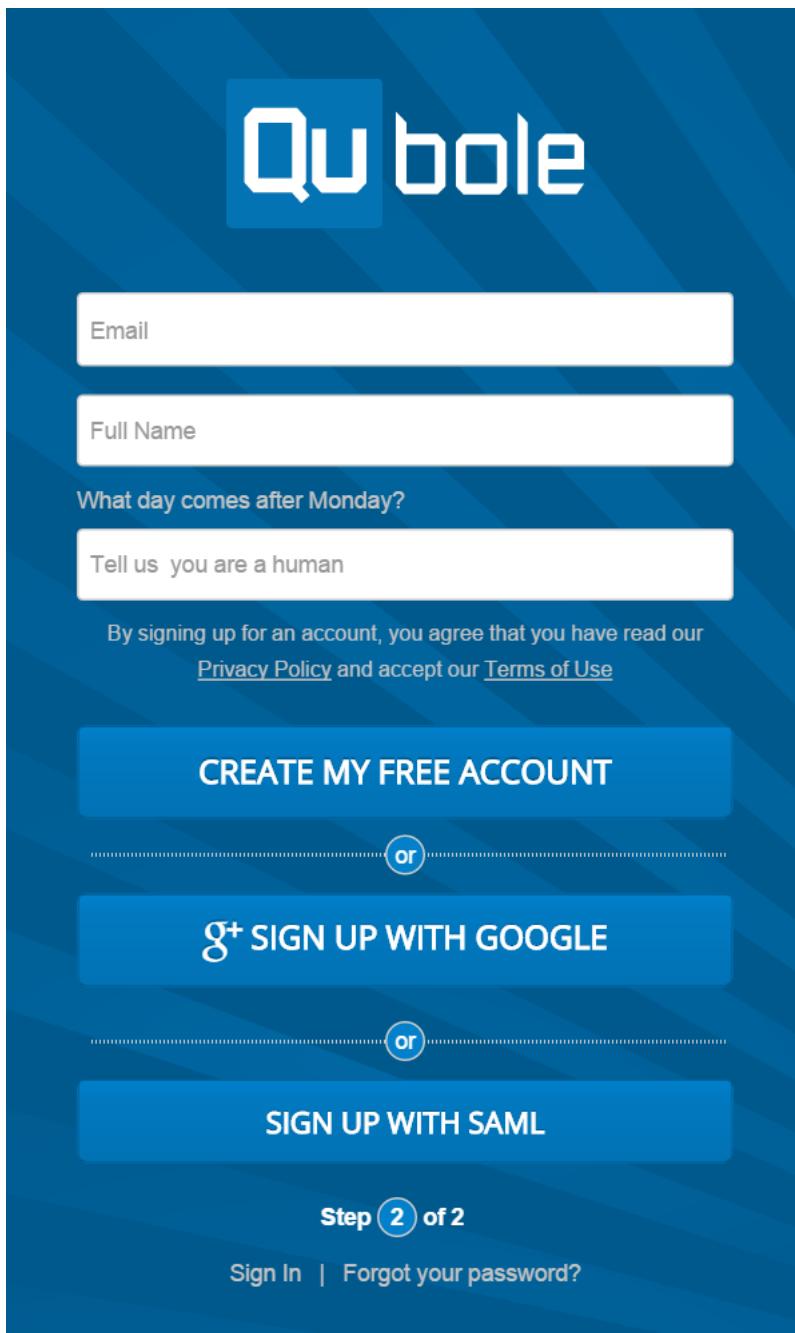
1.1.1 Getting Started with Qubole on AWS

How to Sign up

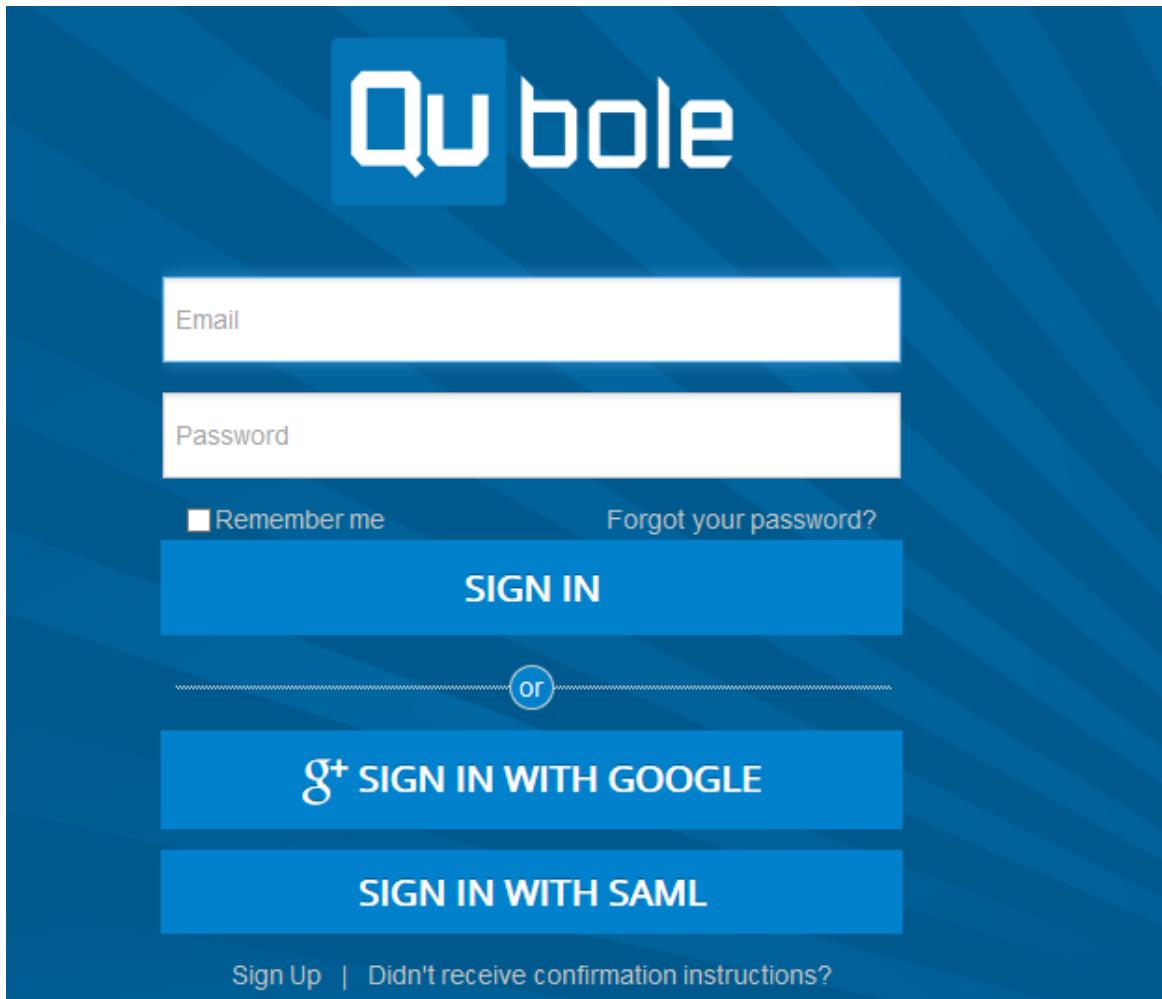
1. Go to <http://www.qubole.com/>.
2. Click **Sign Up for Free**. The following screen is displayed.



3. Provide the required information and click **Next**. The following screen is displayed.



4. Enter your Email ID, Full Name, and answer the simple question. Click **CREATE MY FREE ACCOUNT**. You will receive an email, to the email ID that you provided, with an activation code. You can choose to confirm your account by clicking on the link sent to you in the email or copy and paste the activation code in the signup window. Sign up and you will be provided with a free trial account.
5. After sign up, you can use the following login page to login into Qubole Data Services.



6. Once you login, you will see the **Analyze Page**.

A screenshot of the Qubole Analyze Page. The top navigation bar includes the Qubole logo, Cluster Status, Account dropdown (set to dev-perf-test), and a search bar. The main area features a sidebar with various icons (Compose, History, Repo, Tables, S3) and a search bar containing the query "interface:API,UI,TEMPLATE,ODBC,SMART_QUERY qbol_user_id:veenamj@qubole.com". Below the search bar, a message states "No commands found in last 7 days. Try with 'All' users or broaden your search criteria, or click 'Show More' to view older commands, if any." A "Show More" link is present at the bottom of this section. The rest of the page is mostly blank white space.

7. The **Analyze Page** has the following tabs and a button:

1. The **History** tab lets you retrieve the history of Commands submitted to Qubole Service.
2. The **Compose** button when clicked displays a command editor window that lets you compose and submit tasks.

3. The **Tables** tab lists all the tables on all the databases. Qubole creates a couple of demo tables by default: **default_qubole_airline_origin_dest** and **default_qubole_memetracker**. To view the list of columns on a specific table, expand the table by clicking the arrow to its left.

Column	Type
site	string
ts	string
phr	string
lnks	string
month	string

4. The **S3** tab allows you browse through the S3 folder structure.

How to run your first Hive query, extract sample rows, and analyze data

1. In the **Analyze Pane**, click the **Compose** button. You will see a command editor on the right side, with a list of commands that you can create. By default, **Hive Query** is selected.

2. To run a Hive query, ensure that the **Hive Query** is selected from the drop-down list. In the **Compose** editor window, type a simple query. For example:

```
show tables;
```

3. Click **Run** to execute the query.
4. The Results are shown in the **Results** tab.

Note: If the query is successful, the Log area shows the status of the query as OK and displays the time taken to run the query. Also, next to the Query, a green dot indicates that the query **Succeeded**. You can also click on the History tab to see the query status.

5. To execute another query, click **Compose**. This clears the command window.
6. Now type and execute any other query in the **Compose** editor window. For example:

```
select * from default_qubole_memetracker limit 10;
```

7. Click **Run** to execute the query.

Note: The query takes a little time if a large amount of data has to be fetched.

8. To analyze the data, for example to find the total number of rows in a table corresponding to August 2008, submit the following query:

```
select count(*) from default_qubole_memetracker where month="2008-08";
```

Note: This query is more complex than the previous queries and requires additional resources. In the background, Qubole Data Service provisions a Hadoop cluster.

This can take a couple of minutes. When the query is being processed, the status of the query is shown as **In Progress** with a spinning circle. Once it is processed successfully, the status changes to **Succeeded**.

Congratulations! You have just executed your first Hive query on the Qubole Data Service. Drop an email to help@qubole.com and one of our engineers will get back to you and help you onboard.

1.2 Running a Hadoop Job

This Quick Start Guide is for users who want to run native MapReduce jobs written in Java using Qubole Data Service (QDS).

1.2.1 QDS Access

- Users need to be signed up for QDS. New users can sign up using the [Sign Up](#) page and create an account as instructed
- To run Hadoop jobs using UI, the user must [sign in](#) to QDS.
- To run the Hadoop jobs using the API, the users must have an authentication token that they can access from their profile page. For more information on authentication, see [Authentication](#).

1.2.2 Example Hadoop Job

For this example, we will use a [widely referenced Python Map-Reduce Tutorial](#). The input data set for this job is text from 3 books from Project Gutenberg. The map and reduce programs are python scripts that are used to calculate word counts in this data set. In order to make this example easily accessible to Qubole users, all the required data and code is provided in a publicly accessible bucket:

- **Input Data:** s3n://paid-qubole/default-datasets/gutenberg
- **Map Script:** s3n://paid-qubole/HadoopAPIExamples/WordCountPython/mapper.py
- **Reduce Script:** s3n://paid-qubole/HadoopAPIExamples/WordCountPython/reducer.py
- **Jar File:** s3://paid-qubole/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar This is a standard hadoop streaming jar that is compatible with the Qubole Hadoop service and can be used for all streaming jobs.

1.2.3 Running Hadoop Jobs from Analyze

The steps to run a Hadoop job are:

1. Navigate to the [Analyze](#) page from the top menu and click the **Compose** button.
2. Clicking **Compose** opens a command editor. Select the command type as **Hadoop Job** from the drop-down list.
3. Specify the location of the job JAR file (in this case: s3://paid-qubole/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar)
4. Specify the arguments to the JAR file. In the example shown below, specify the mapper/reducer scripts, the location of these scripts, the number of reducers and the location of the input dataset, and an output S3 bucket location as arguments that are as shown below.

```
-files s3n://paid-qubole/HadoopAPIExamples/WordCountPython/mapper.py,
s3n://paid-qubole/HadoopAPIExamples/WordCountPython/reducer.py
-mapper mapper.py -reducer reducer.py -numReduceTasks 1
-input s3n://paid-qubole/default-datasets/gutenberg
-output s3n://.../...
```

Note: The output path shown in the above step and the following figure is not an actual path. Provide an output location in an S3 bucket that you own.

Figure: Sample Hadoop Job

5. Click **Run** to execute the job. The status of the job is displayed on the top of the query composer. The query result is displayed in the **Results** tab.

Viewing Hadoop Job Logs

Upon clicking **Submit**, the job progress can be monitored by viewing the logs. The job submission logs are available under the Log section in the Composer tab and also in the History tab for later access. The job log provides a Job Tracker URL, which when clicked displays detailed information of the job, such as map and reduce task information.

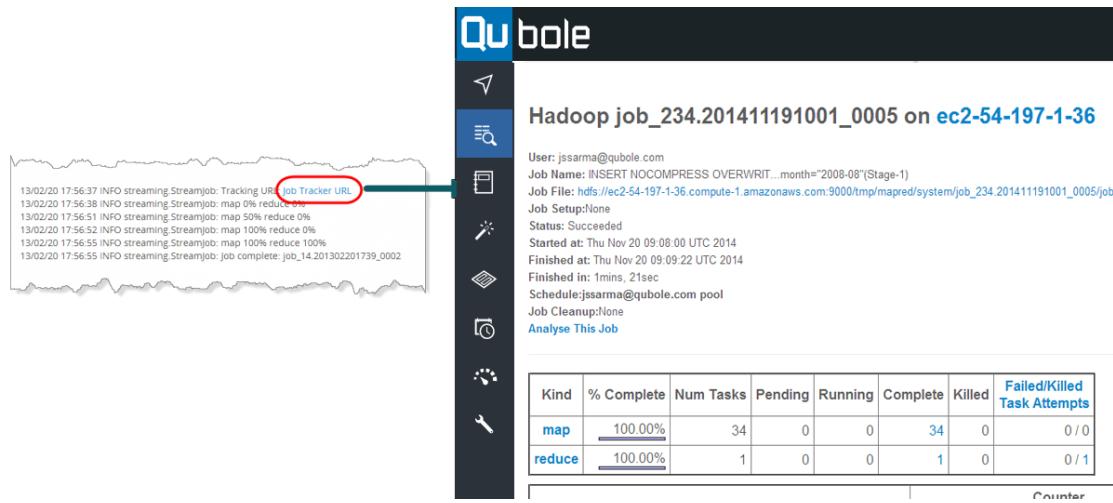


Figure: Sample Hadoop Log

Congratulations! You have executed your first Hadoop command using Qubole Data Service.

1.2.4 Running Hadoop Jobs using the API

Hadoop jobs can also be run from the command line using [Qubole API interface](#). The following steps show how this can be accomplished. **Note:** The environment variable, *AUTH_TOKEN* in these examples, must be populated using the authentication token of the user as described [Authentication](#).

1. Submit the command:

```
unix-prompt > curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -d '{ "id": 1, "name": "John Doe", "email": "john.doe@example.com", "password": "password123" }' https://api.example.com/users
```

HTTP/1.1 200 OK

```
{"status":"waiting","qbol\_session\_id":30867,"created\_at":"2013-04-22T11:37:32Z","command\_type": "run","files": "s3n://paid-qubole/HadoopAPIExamples/WordCountPython/mapper.py,s3n://paid-qubole/HadoopAPIExamples/reducer.py -numReduceTasks 1 -input s3n://paid-qubole/default-datasets/gutenberg -output s3://bucket.data.com/qubole/grun2}","resolved\_macros":null,"progress":0,"qlog":null,"path":"/tmp/wordcountjob_30867","queue": "Hadoop API Examples","region": "us-east-1","status": "waiting","submitted_at": "2013-04-22T11:37:32Z","version": 1}
```

The **ID** of the command is *137222* as shown in the result of the REST API call. Let us use this ID to check for the status of the command.

- ## 2. Check Status of command:

```
unix-prompt > curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json"
```

HTTP/1.1 200 OK

```
{"status":"done","qbol\_session\_id":30867,"created\_at":"2013-04-22T11:37:32Z","command":{"job\_files  
s3n://paid-qubole/HadoopAPIExamples/WordCountPython/mapper.py,s3n://paid-qubole/HadoopAPIExamples/mapper_mapper.py -reducer reducer.py -numReduceTasks 1 -input s3n://paid-qubole/default-datasets/gutenberg -output s3://bucket.data.com/qubole/grun2},"meta\_data":{"results\_resource":"commands/137222/results"}}
```

The status field shows whether the command is *waiting*, *running*, *done*, and so on. In this case, the command has already been completed (status is *done*).

3. Get the logs of the command:

```
unix-prompt > curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json"
```

Further documentation is available at our [Documentation home page](#).

1.3 Running a Hive Query

This document is intended to get a new user up and running with QDS, by running a simple Hive query. As a prerequisite, the user must have signed up for Qubole (via [Sign Up](#) page) and has a working account in QDS (else, create one [here](#)).

1.3.1 Step I: Explore Tables

Navigate to the [Analyze](#) page from the top menu. Click the **Tables** tab. It shows the list of databases.

1. Click the database to view the list of all the tables in it.
2. All accounts have access to two pre-configured tables in the default database: **default_qubole_airline_origin_destination** and **default_qubole_memetracker**.
3. To view the list of columns of a specific table, click on the arrow sign to the left of the table name (see image below).

The screenshot shows the Qubole Analyze interface with the 'Tables' tab selected. The 'default_qubole_memetracker' table is expanded, showing its columns and types:

Column	Type
site	string
ts	string
phr	string
lnks	string
month	string

Below the table, there are links to other tables: demo_data1, demo_data3, and demo_memetracker.

Figure: Pre-Configured Tables

1.3.2 Step II: View Sample Rows

Now, execute a simple query against this table by entering the following text in the query box:

```
select * from default_qubole_memetracker limit 10
```

Click **Run**. Within a few seconds, you should see 10 rows from the table show up in the **Results** tab.

site	ts	phr	links	month
http://codeproject.com/kb/silverlight/convertsilverlightcontrol.aspx	2008-08-01 00:00:00	["how to create property binding in a visual webgui silverlight control","videoplayer silverlight controls videoplayer videoplayer silverlight controls version 1 0 0 0 culture neutral publickeytoken null","videoplayer controls videoplayer videoplayer controls"]		2008-08
http://wallstreetexaminer.com/?p=2987	2008-08-01 00:00:01	[]		2008-08
http://news.bbc.co.uk/go/rss-/1/hi/scotland/highlands_and_islands/7535558.stm	2008-08-01 00:00:01	["our continuing strategic priority is to provide a safe and efficient group of airports while pursuing development opportunities which improve the air transport network serving the region","our results for the year demonstrate that		2008-08

Figure: View Some Rows

1.3.3 Step III: Analyze Data

To get the total number of rows in the table corresponding to August, 2008. Use the following query:

```
select count(*) from default_qubole_memetracker where month="2008-08".
```

This query is more complex than the previous one and requires additional resources. Behind the scenes, Qubole Data Service provisions a Hadoop cluster, which may take a couple of minutes. The provisioning of the Cluster is indicated with the icon:

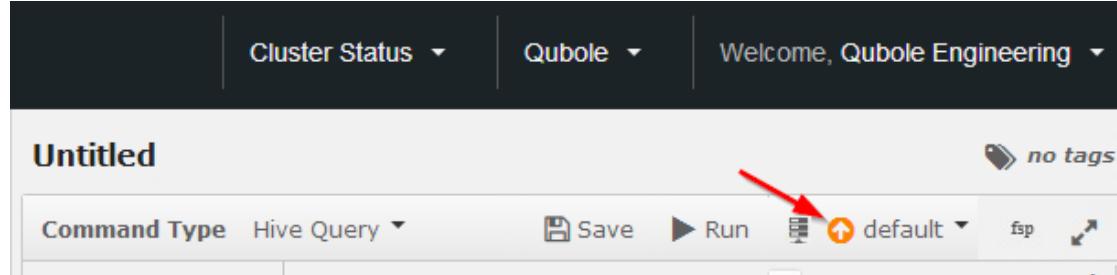
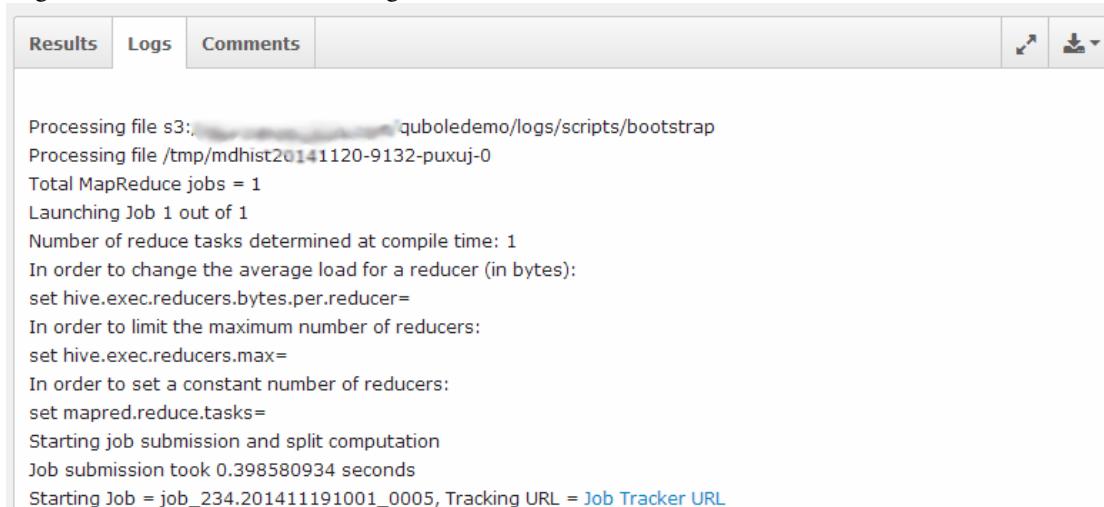


Figure: Cluster provisioning

Once the cluster is provisioned, you see the query's progress in the **Log** tab.

Finally, you see the logs and results below the query composer.

Logs section looks like the following illustration:



The screenshot shows a user interface with a top navigation bar containing 'Results', 'Logs' (which is selected), and 'Comments'. There are also icons for sharing and download. The main area displays the following log output:

```

Processing file s3://quboledemo/logs/scripts/bootstrap
Processing file /tmp/mdhist20141120-9132-puxuj-0
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=
In order to set a constant number of reducers:
set mapred.reduce.tasks=
Starting job submission and split computation
Job submission took 0.398580934 seconds
Starting Job = job_234.201411191001_0005, Tracking URL = Job Tracker URL

```

Congratulations! You have executed your first query using the Qubole Data Service.

Further documentation is available at our [Documentation home page](#).

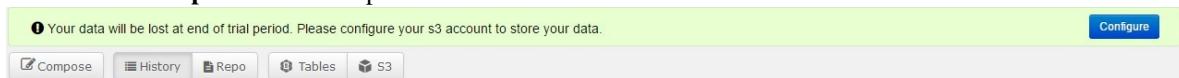
1.4 Running Spark Applications

This document is intended to guide a new user to run Spark applications on QDS. As a prerequisite, you must have signed up for Qubole (via [Sign Up](#) page) and have a working account in QDS (else, create one [here](#)). To run Spark Applications from the QDS UI, you must [sign in](#) to QDS.

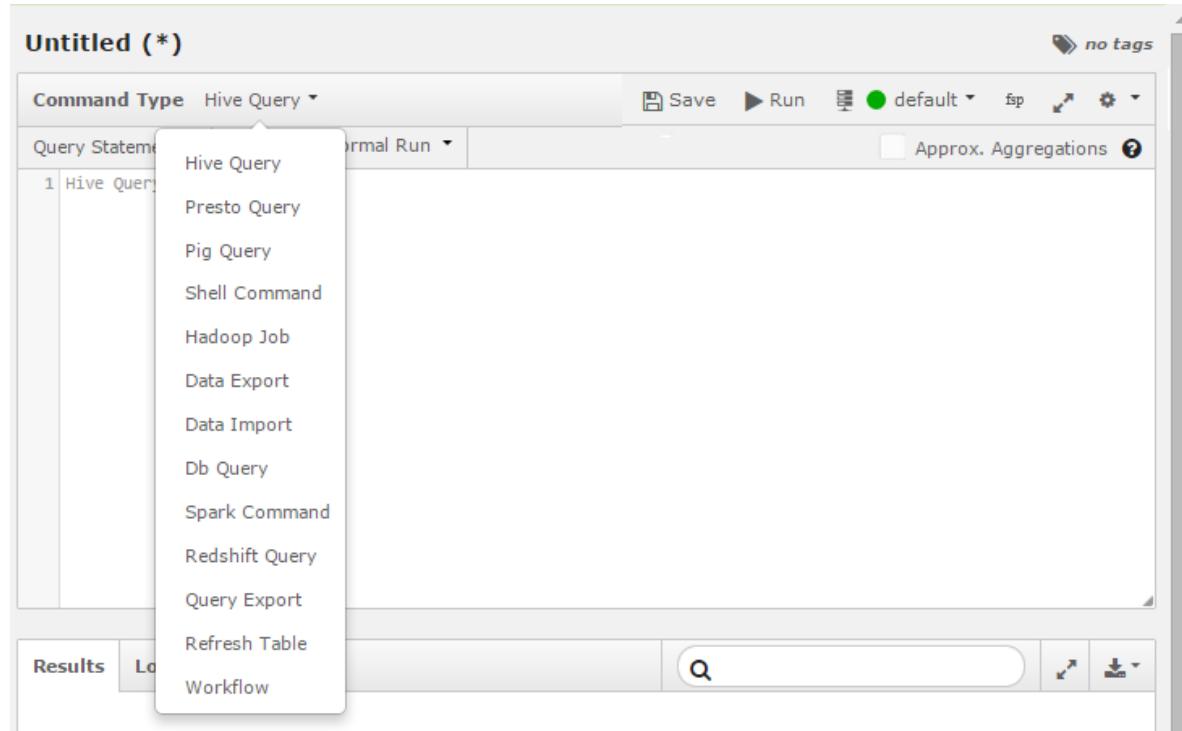
1.4.1 Submitting a Spark Application

After signing in, you see the [Analyze](#) page. This page shows all the queries that are run so far in the **History** tab. For a first-time sign in, **History** contains some example commands. The steps given below show how to run a Spark command:

1. Click **Compose** from the top menu.



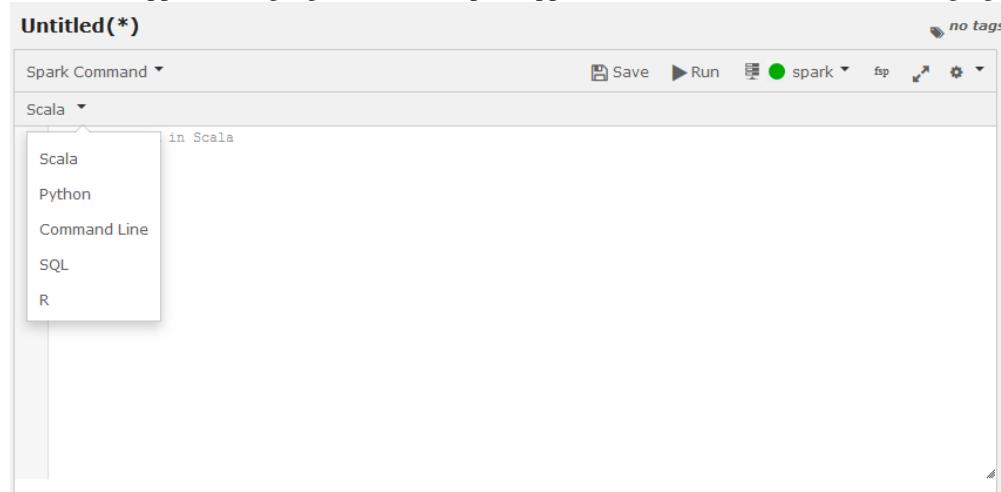
2. Select **Spark Command** from the **Command Type** drop-down list.



3. You see an editor that can be used to write a Scala Spark application. Qubole also allows you to change the language setting to write a *Python*, *Command-line*, *SQL*, or *R* Spark application. See the following topics for more information:

- [Compose a Spark Application in Command Line](#)
- [Compose a Spark Application in Python](#)
- [Compose a Spark Application in Scala](#)
- [Compose a Spark Application in SQL](#)
- [Compose a Spark Application in R](#)

The supported languages to write a Spark application are as shown in the following figure.



4. Let us start with a simple Scala Spark application. Type the following text into the editor.

The screenshot shows the Qubole Data Service (QDS) interface. At the top, there is a code editor window titled "Untitled (*)" containing the following Scala code:

```

import org.apache.spark._
object FirstProgram {
  def main(args : Array[String]) {
    val sc = new SparkContext(new SparkConf())
    val result = sc.parallelize(1 to 10).collect()
    result.foreach(println)
  }
}

```

Below the code editor is a toolbar with "Save", "Run", and other options. To the right of the toolbar is a dropdown menu labeled "default" which lists four cluster options: "default" (green), "hadoop2" (red), "presto" (red), and "spark" (green). A "no tags" icon is also present.

Underneath the toolbar, there is a section titled "Spark Submit Command Line Options" containing the command: "--class FirstProgram".

- Execute this program by clicking **Run** at the top of the editor. **Congratulations!** You have run your first Spark application using QDS.

By default, Qubole provides some configured clusters to each account, which includes a Spark cluster. Qubole manages the complete lifecycle of the cluster. When you submit a Spark command, it automatically brings up the Spark cluster. When the cluster is idle for sometime, it terminates the cluster. For more details, see [Introduction to Qubole Clusters](#).

Also, note that the above action brings up the cluster, which takes a few minutes to start. All subsequent queries holds on to the same cluster.

1.4.2 Viewing the Logs and Results

The **Logs** and **Results** tabs are available just below the query composer. The pane that is to the left of the query composer is for **History**, which contains all queries run through Qubole. You can check the logs and results of any previously submitted query.

Note: You can access a Spark job's logs even after the cluster on which it was run is terminated.

Spark Submit Command Line Options

```
--class FirstProgram
```

Arguments for User Program

Results Logs Comments

Loading ...
Map: Reduce: Stage: 1

```
Qubole > 2015-02-14 01:38:05,479 INFO shellcli.py:240 - main - Shell CLI Begin  
2015-02-14 01:38:06,086 INFO cmd_utils.py:66 - _must_get_master_ip - Getting Hadoop cluster information  
Qubole > 2015-02-14 01:38:09,914 INFO shellcli.py:338 - getHadoopShellLauncherCommand -  
['/usr/lib/hadoop2/bin/hadoop', 'jar', '/usr/lib/hadoop/contrib/qubole/hadoop-0.20.1-dev-qubole.jar',  
'com.qubole.ShellLauncher', u'Dmaster.hostname=ec2-54-159-45-246.compute-1.amazonaws.com', '-  
Dhadoop.inh.uu=enselukatnew@qubole.com.default_group' '-Dmanred.inh.name=Shell_Command']
```

1.4.3 Accessing Data Sets in S3

In a cloud-based model, data sets are typically stored in S3 and are loaded into Spark. Here is the familiar example of word count against a public data set accessible from all Qubole accounts.

```
import org.apache.spark._  
import org.apache.spark.SparkContext._  
  
object s3readwrite {  
  
  def main(args: Array[String]) {  
    val sc = new SparkContext(new SparkConf())  
    val file = sc.textFile("s3://paid-qubole/default-datasets/gutenberg/pg20417.txt")  
    val counts = file.flatMap(line => line.split(" "))  
      .map(word => (word, 1))  
      .reduceByKey(_ + _)  
      .collect()  
    counts.foreach(println)  
  }  
}
```

To start accessing data in your S3 buckets, navigate to **Control Panel > Account Settings > Storage Settings** and enter AWS S3 access and secret keys.

1.4.4 Passing Arguments to Spark Application

Let us consider parameterizing the S3 location in the previous application and pass it as an argument. Here is an example of how to do it.

```

import org.apache.spark._
import org.apache.spark.SparkContext._

object s3readwrite {

  def main(args: Array[String]) {
    val sc = new SparkContext(new SparkConf())

    //taking first argument as location in s3
    val file = sc.textFile(args(0))

    val counts = file.flatMap(line => line.split(" "))
      .map(word => (word, 1))
      .reduceByKey(_ + _)
      .collect()
    counts.foreach(println)
  }
}

```

Now, pass the S3 location in the **Arguments for User Program** text box. Try the same public location, **s3://paid-qubole/default-datasets/gutenberg/pg20417.txt**. In this way, the program can take in any number of arguments.

1.4.5 Using Notebooks

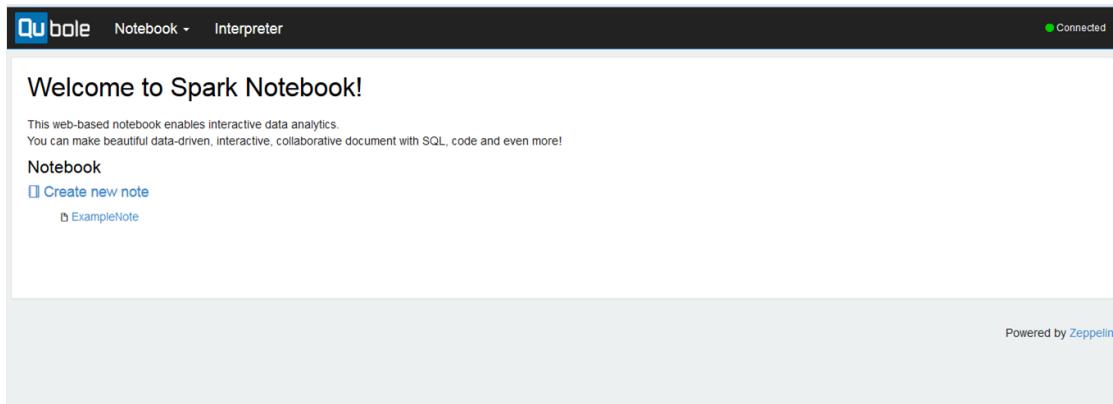
To run Spark applications in Notebook style interface, follow this quick start guide, [Running Spark Applications in Notebooks](#).

1.5 Running Spark Applications in Notebooks

Log-in to **Qubole** with your username/password. Go to **Control Panel** page and start the Spark cluster. Once the cluster is up, Spark notebook link shows up on the UI.

Active Cluster(s)		Deleted Cluster(s)			
		Search : Enter Search Text			
Id	Labels	Nodes	Up Time	Resources	Action
● 16866 ✓	default	0		Cluster Usage Report	
● 16867	presto	0		Cluster Usage Report	
● 16868	hadoop2	0		Cluster Usage Report	
● 16869	spark	2 (0)	6 minutes	Cluster Usage Report Resource Manager AutoScaling-Logs View Master DNS	Spark Notebook

Click **Spark Notebook** and it opens the Notebook in a new page.

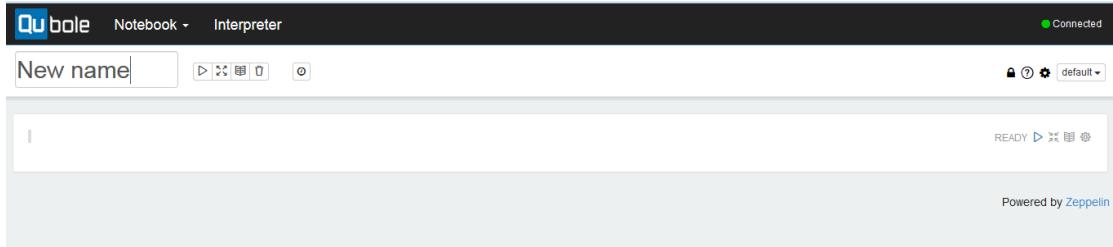


The green dot on top right (next to **Connected**) indicates that web socket is connected. Default idle timeout of web-socket is 10 minutes. This page comes with a Example Notebook to give user an overview of supported functionalities.

1.5.1 Notebooks

You can create any number of new notebooks. Notebooks data is synced and persisted in S3 regularly. Notebooks are associated with cluster, so notebooks from cluster A can not be accessed by cluster B.

Qubole Spark Notebooks come with an Example notebook to showcase various features. It has examples for Spark Pi program, word count, show tables on hive metastore and so on. Notebook names can be changed on the Notebook page.



With Qubole First Class Notebooks, you now have the ability to lock and unlock notes. When you lock a note, you can prevent edits on the note from other users in the account. Once the notebook is ready to be used, you can unlock the note to make it available to all users in the account. This simplifies the user experience in a multiuser Qubole account.

To lock a notebook, click the lock icon, .

When you lock a note, the icon turns unlocked that indicates that note is locked as illustrated in the following figure.

Introduction

```
%md ##This notebook supports multiple language backend
- default: scala with SparkContext available as sc
- %md: markdown
- %sql: hive on spark by default. hiveContext object is also available by default. If you specify zeppelin.spark.useHiveContext as false in interpreter settings then this will use spark sql and sqlContext object becomes available
- %pyspark: pyspark
- %sh: shell

Interpreters are loaded lazily and hence takes some time the first time.
Note uses a websocket for communication. So changes propagate to all open instances of this notebook instantaneously.
User can configure parameters with which sc object is created by going to interpreters.
User can create as many sc interpreters as they want with different settings
Qubole Notes supports auto assist for code completion. You can view the suggestions using the key combination __Ctrl + .(period)__

#Note that
- If you use sc.stop in the notebook then spark context will stop and you have to go to interpreter page and click on restart to restart that interpreter again.
```

By locking a note, you get exclusive control over it. Other users in the account can only view the note and cannot edit or delete the note until you unlock it. For unlocking a note, click the unlocked icon in the locked note as illustrated in the following figure.

Introduction

```
%md ##This notebook supports multiple language backend
- default: scala with SparkContext available as sc
- %md: markdown
- %sql: hive on spark by default. hiveContext object is also available by default. If you specify zeppelin.spark.useHiveContext as false in interpreter settings then this will use spark sql and sqlContext object becomes available
- %pyspark: pyspark
- %sh: shell

Interpreters are loaded lazily and hence takes some time the first time.
```

1.5.2 Notebook Interpreters

Notebooks support Python, Spark, SQL, markdown, and shell as interpreters. The procedure to use various interpreters is explained in Example Notebook. You can define custom settings for Spark interpreter by clicking **Interpreter** link on top. Any number of interpreter setting objects can be created.

Properties	Value
name	
args	
master	yarn-client
spark.app.name	Zepelin
spark.executor.cores	1
spark.executor.instances	2
spark.executor.memory	1g
spark.qubole.idle.timeout	60
spark.qubole.max.executors	10
zeppelin.spark.maxResult	10000

You can specify any Spark settings. Spark interpreter started by notebooks have the specified settings. When a new Spark interpreter is created, some settings are shown by default with description. You can change these settings as

required. One of important setting is `spark.qubole.idle.timeout`. This setting is number of minutes after which Spark context shuts down if no job has run in that Spark context.

name	value	description	action
args		spark commandline args	x
master	yarn-client	spark master url. ex) spark://masterhost:7077	x
spark.app.name	Zepplin	The name of spark application	x
spark.executor.cores	1	total number of cores to use per executor	x
spark.executor.instances	2	Min executors to start spark application with	x
spark.executor.memory	1g	executor memory per worker instance	x
spark.qubole.idle.timeout	60	spark context will stop automatically if it has not run anything since last spark.qubole.idle.timeout minutes. Set to -1 to never stop spark. This is necessary to free up cluster resources	x
spark.qubole.max.executors	10	Max executors autoscaling can reach to	x

About Spark SQL Command Concurrency

In notebooks, you can run multiple Spark SQL commands in parallel. This is controlled by setting `zeppelin.spark.concurrentSQL` to true. The exact value of concurrency is controlled by `zeppelin.spark.sql.maxConcurrency`, which is set to a positive integer. The default value of this parameter is 10.

1.5.3 Associating Interpreters with Notebooks

Creating an interpreter setting does not start interpreter. Interpreters are started on demand when required by notebooks. You have to associate an interpreter setting with a notebook so that it can start Spark context with those settings.



You can see the available interpreter settings by clicking the **Gear icon** on top left and associate any interpreters with notebook by clicking the interpreter settings name.

You have to click **Save** to get the commands on notebook working.

1.5.4 SC Object

Similar to a Spark shell, a SC object is available in the Notebook. Along with this, hiveContext is also available in a Notebook if you have to set the following configuration to true while creating interpreter settings:

```
zeppelin.spark.useHiveContext
```

If this configuration is not set to true, then sqlContext is available in a Notebook in place of hiveContext.

1.6 Running a Pig Job

This Quick Start Guide is for users who want to run Pig jobs using Qubole Data Service(QDS). To run the Pig jobs using our API, you must have a working account in QDS. If you do not have one, click [here](#) to create an account using sign up.

1.6.1 Files Used in the Demo

Pig script and the UDFs must be uploaded to Amazon S3. For quick start, we have already uploaded some sample Pig scripts and UDFs to our S3 Bucket, s3://paid-qubole/PigAPIDemo. Here are the list of files:

1. **/data/excite-small.log** - The dataset used to crunch.
2. **/jars/tutorial.jar** - Pig provides extensive support for user-defined functions (UDFs) as a way to specify custom processing. We have uploaded a sample Java UDF here.
3. **/scripts/script1-hadoop-s3-small.pig** - This is the pig script we will be running.
4. **/scripts/script1-hadoop-parametrized.pig** - This script is same as the above except that its parametrized. It takes in the following as parameters – \$input,\$output and \$udf_jar

The above script is Query Phrase Popularity script which processes a search query log file from the Excite search engine and finds search phrases that occur with particular high frequency during certain times of the day. These files are cloned from the [Apache Pig Tutorial](#) and it has an explanation of what this script does.

1.6.2 Steps

Step 1: Get the Access Token

Get the API access token as mentioned [Authentication](#).

Set the environment variables:

1. `export AUTH_TOKEN={your account's auth token}`
2. `export V=v1.2 # api rev at the time of this writing`

Step 2: Submit a Pig Command

Let us submit the non-parametrized Pig Script using Curl.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

It will return a JSON response of command object. Take note the **id** in the JSON response.

Similarly, to submit the parametrized script, run this command:

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Step 3: Get the Command Status

From the JSON response, get the “id”. To check the status of the Pig Command – replace the \${id} in the request below with the actual value from the response.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Step 4: Check the Logs and Results

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Congratulations! You have submitted the first Pig job.

You can also provide the PigLatin statements in the request itself. For more details, refer the API documentation [Submit a Pig Command](#)

1.7 Running a Shell Command

This document is intended to get a new user up and running with QDS, by running a simple Shell Command. As a prerequisite, the user must have signed up for Qubole (via [Sign Up](#) page) and has a working account in QDS (else, create one [here](#)).

1.7.1 Running a Shell Command from Analyze

The steps to run a Shell Command are:

1. Navigate to the [Analyze](#) page from the top menu and click the **Compose** button. A command composer and editor is displayed.
2. In the **Compose** command editor, select the command type as **ShellCommand** from the drop-down list.
3. Specify the shell command to be run. (For example, `hadoop dfs -ls s3://paid-qubole`).
4. Optionally specify other files or archives to be copied to the directory where the shell command executes.
5. Click **Run** to execute the job. The status of the job is displayed on the top of the **Compose** command editor.

The following figure shows a sample shell command in the **Compose** command editor of the [Analyze](#) page.

The screenshot shows the Qubole Data Service interface. At the top, there are tabs for "Shell Command" and "Bash Commands", with "Shell Command" selected. Below the tabs is a toolbar with icons for Save, Run, and other options. The main area contains a command line input field with the command "hadoop dfs -ls s3://paid-qubole". Below the command line, there are two sections for optional file and archive inputs. The first section is for files, with the placeholder "Comma separated list of files to be copied". The second section is for archives, with the placeholder "Comma separated list of files to be copied". Both sections have a note below them stating they will be copied to the working directory where the command is executed.

Congratulations! You have executed your first shell command using the Qubole Data Service.

Further documentation is available at our [Documentation home page](#).

1.8 Running a Mahout Job

This Quick Start Guide is for users, who want to run Mahout jobs using Qubole Data Service (QDS).

1.8.1 Example Mahout Job

For this example, a [simple recommender job](#) is used. To make this example easily accessible to Qubole users, the required data and code are provided in a publicly accessible bucket:

- **Input Data:** s3://paid-qubole/mahout/links-converted.txt and s3://paid-qubole/mahout/users.txt
- **Jar File:** s3://paid-qubole/mahout/mahout-core-0.7-job.jar This is the Mahout jar version 0.7.

1.8.2 Running Mahout Jobs from Analyze

Perform the following steps to run a Mahout job:

1. Navigate to the [Analyze](#) page from the top menu and select the **Compose** tab.
2. In **Command Type**, select the command type as **Hadoop job** from the drop-down list.

3. Specify the location of the job JAR file in the **Path to Jar File** text field (in this case: s3://paid-qubole/mahout/mahout-core-0.7-job.jar)
4. Specify the arguments to the JAR file in the **Arguments** text field. In the illustrated example provided below, these are the arguments:

```
org.apache.mahout.cf.taste.hadoop.item.RecommenderJob  
-Dmapred.input.dir=s3://paid-qubole/mahout/links-converted.txt  
-Dmapred.output.dir=hdfs:///tmp/mo1 --usersFile  
s3://paid-qubole/mahout/users.txt --booleanData -s SIMILARITY_LOGLIKELIHOOD  
--tempDir hdfs:///tmp/mo1-inter
```

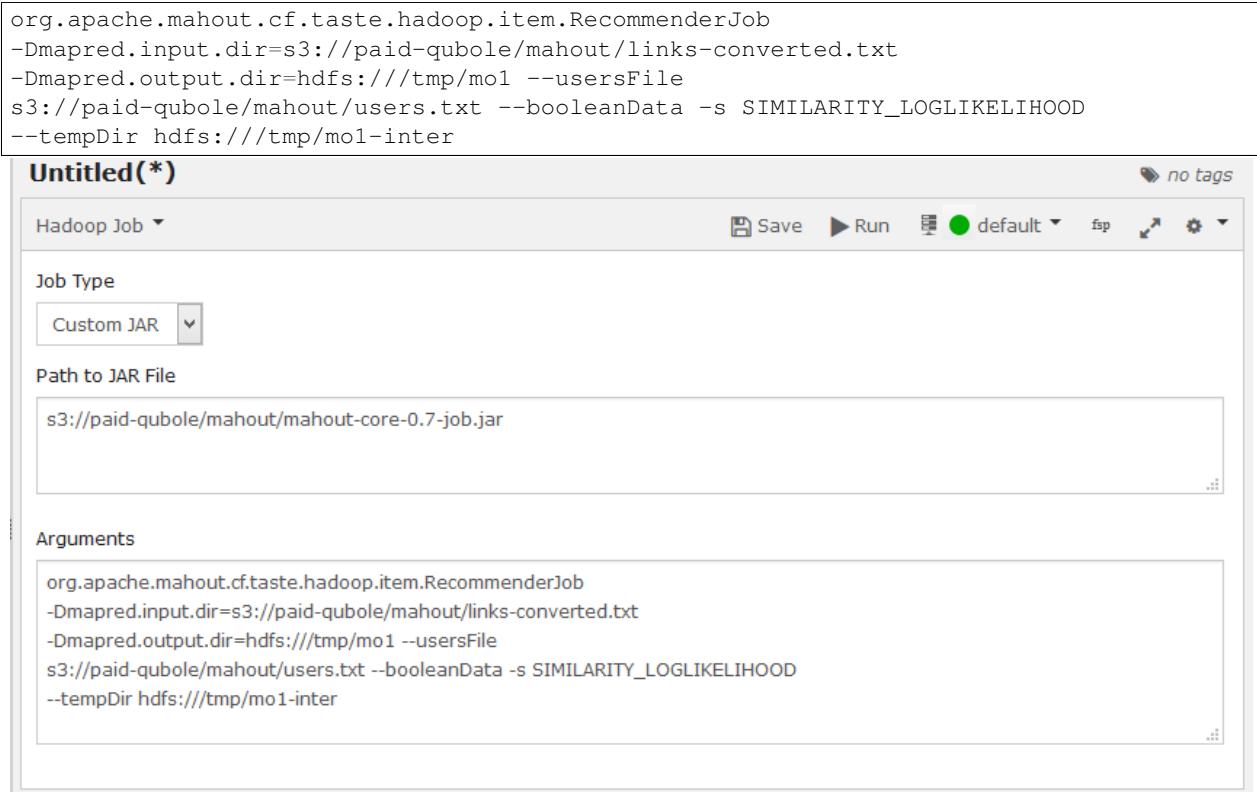
Untitled(*) no tags

Hadoop Job Save Run default fsp

Job Type Custom JAR

Path to JAR File
s3://paid-qubole/mahout/mahout-core-0.7-job.jar

Arguments
org.apache.mahout.cf.taste.hadoop.item.RecommenderJob
-Dmapred.input.dir=s3://paid-qubole/mahout/links-converted.txt
-Dmapred.output.dir=hdfs:///tmp/mo1 --usersFile
s3://paid-qubole/mahout/users.txt --booleanData -s SIMILARITY_LOGLIKELIHOOD
--tempDir hdfs:///tmp/mo1-inter



5. Click **Run** to execute the job. The status of the job is displayed in the **Results** tab.

Note: You can provide an output location in a bucket that you own.

Congratulations! You have executed your first Mahout command using QDS.

You can also run a Mahout job for the example mentioned above by running a shell command. In the query composer of the [Analyze](#) page, select **Shell Command** from the **Command Type** drop-down list. Enter the bash command, hadoop dfs -cat /tmp/mo1/part* in the **Bash Commands** text field. Click **Run** to execute the job.

Further documentation is available at our [Documentation home page](#).

1.9 Running a Dumbo Job

[Dumbo](#) is a popular Python module for running Hadoop jobs. This Quick Start Guide is for users who want to run Dumbo programs using Qubole Data Service (QDS).

1.9.1 Example Dumbo program

For this example, let us use the canonical word-count program, but use Dumbo instead of plain Python. To make this example easily accessible to Qubole users, the required data has been provided in a publicly accessible bucket, and the Python program as a publicly accessible pastebin paste (since Dumbo does not work directly with s3 files).

- **Input Data:** `s3://paid-qubole/default-datasets/gutenberg`, which contains a small subset of books from Project Gutenberg
- [Dumbo program](#) (link to raw file)

1.9.2 Installing Dumbo

The simplest way to install Dumbo on a cluster is to do so in its *node bootstrap file*. Add the following line in the bootstrap file:

```
easy_install -z dumbo
```

This will install the required modules on all the cluster nodes, so Qubole's command infrastructure may be used to run the programs.

1.9.3 Running Dumbo Jobs from Analyze

The steps to run a Dumbo job are:

1. Navigate to the [Analyze](#) page from the menu on the left and click on the **Compose** button.
2. Select the command type as **ShellCommand** from the drop-down list.
3. In the editor window, provide the following commands in order:

```
wget -q http://pastebin.com/raw.php?i=8RVaucJf -O /tmp/wordcount.py
dumbo start /tmp/wordcount.py -input s3://paid-qubole/default-datasets/gutenberg -output /dumbo/
dumbo cat /dumbo/wc-output -hadoop /usr/lib/hadoop | sort -k2nr | head -10
```

4. Click **Run** to execute the job. The job will return the top 10 words from the sample data set. The progress of the jobs may be monitored in the **Logs** tab below.
5. Once the job is completed, the results will be available in the **Results** tab, as below:

Results	Logs	No Comments	<input type="text"/>		
Colo ▾					
the 42074					
of 23935					
and 16904					
a 12064					
to 12017					
in 11867					
is 7401					
that 6109					
it 4981					
with 4686					

Fig. 1.1: Results from Dumbo wordcount job (not surprisingly, 'the' is right on top)

Congratulations! You have executed your first Dumbo program using Qubole Data Service.

Further documentation is available at our [Documentation home page](#).

1.10 Uploading a File to Amazon S3 Buckets

This document is intended to guide a new user to upload a file into Amazon S3 buckets.

1.10.1 Enable File Upload Setting in the Control Panel

Navigate to **Control Panel** and select **Allow Upload to S3** listed below **Storage Settings** in the **Account Settings** tab. See [Managing Account Settings](#) for more information. You can also select **Allow Download from S3** to enable downloading files. The options are as shown in the following figure.

Storage Settings	
Access Key	AKIATMTATMTATMTA
AWS Secret Key	GXXCMs6/as6/as6/as6/as6/as6/as6/as6/as6/asJ6r
Default Location (for the created data)	data.com/user_hu_481 ⓘ
S3 Cache Size	25 GB
Allow download from S3	<input type="checkbox"/>
Allow upload to S3	<input checked="" type="checkbox"/>
Save Reset	

1.10.2 Configure CORS Policy on a S3 Bucket

You must configure the Cross Origin Resource Sharing (CORS) policy on a S3 bucket to which you want to upload a file.

If you are a new user, you can configure the CORS policy on the default storage S3 bucket that is set in **Control Panel**. Click the [Account Settings](#) tab in the **Control Panel** page.

Copy the default location, mentioned against the **Default location (for the created data)** that is listed below **Storage Settings**.

Perform these steps to configure CORS policy on a S3 bucket:

1. Login to Amazon S3.
2. Navigate to the default bucket, **prod.quoble.com**.
3. Go to **Permissions** and click **Add CORS Configuration**. Set the CORS policy as shown in the following figure.

Bucket: prod.qubole.com X

Bucket: prod.qubole.com
 Region: US Standard
 Creation Date: Mon Oct 13 11:38:23 GMT+530 2014
 Owner: Me

▼ Permissions

Grantee: List Upload/Delete View Permissions X

Edit Permissions

Add more permissions
 Add bucket policy
 Add CORS Configuration

4. Repeat steps 2-3 to set the CORS policy on the S3 buckets to which you want to upload files.

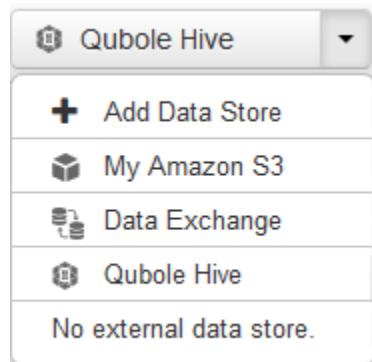
A sample CORS configuration XML file is as shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration>
    <CORSRule>
        <AllowedOrigin>https://api.qubole.com</AllowedOrigin>
        <AllowedMethod>GET</AllowedMethod>
        <AllowedMethod>POST</AllowedMethod>
        <MaxAgeSeconds>3000</MaxAgeSeconds>
        <AllowedHeader>Authorization</AllowedHeader>
    </CORSRule>
</CORSConfiguration>
```

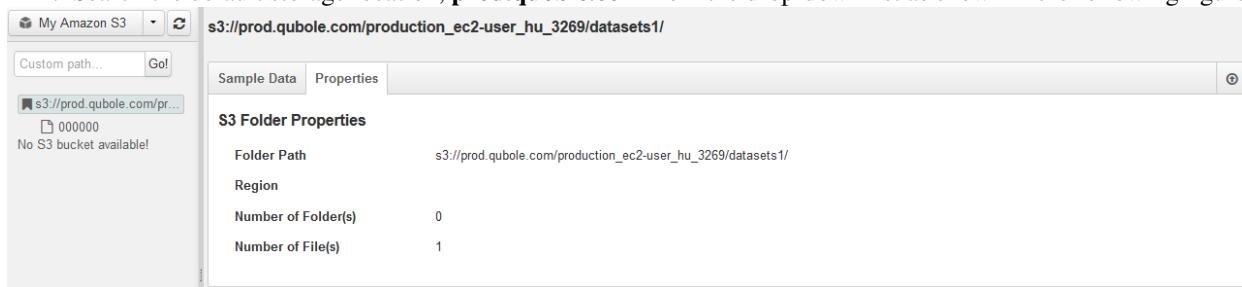
1.10.3 Upload a File into a S3 Bucket Using Qubole UI

Perform these steps to upload a file into a S3 bucket using Qubole UI:

1. Navigate to the [Explore](#) page, click **My Amazon S3** from the drop-down list as shown in the following figure.

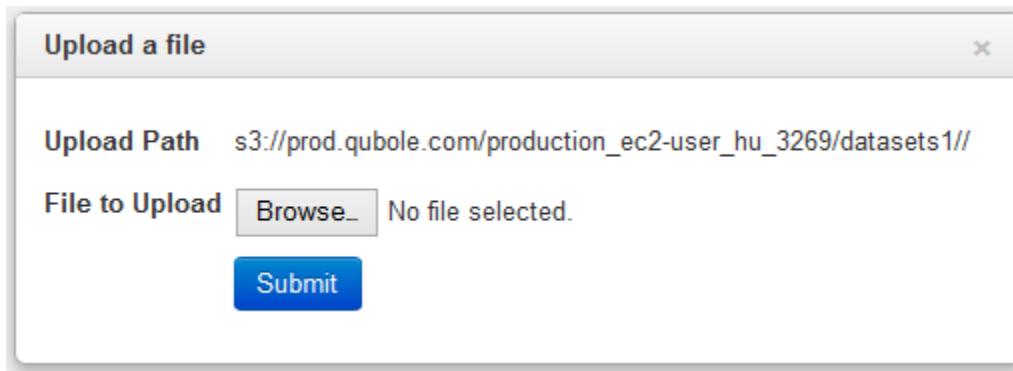


2. Search the default storage location, **prod.quoble.com** from the drop-down list as shown in the following figure.



As a prerequisite, the CORS policy has to be set on the **prod.quoble.com** bucket.

3. Click the file upload icon and the file upload dialog is displayed as shown in the following figure.



4. Click **Browse** to the location of the file and select the file. Click **Submit**. A success message is displayed if the upload is successful. An error message is displayed if the upload is unsuccessful.

Go to the corresponding S3 bucket and check to confirm the uploaded file listed in the bucket.

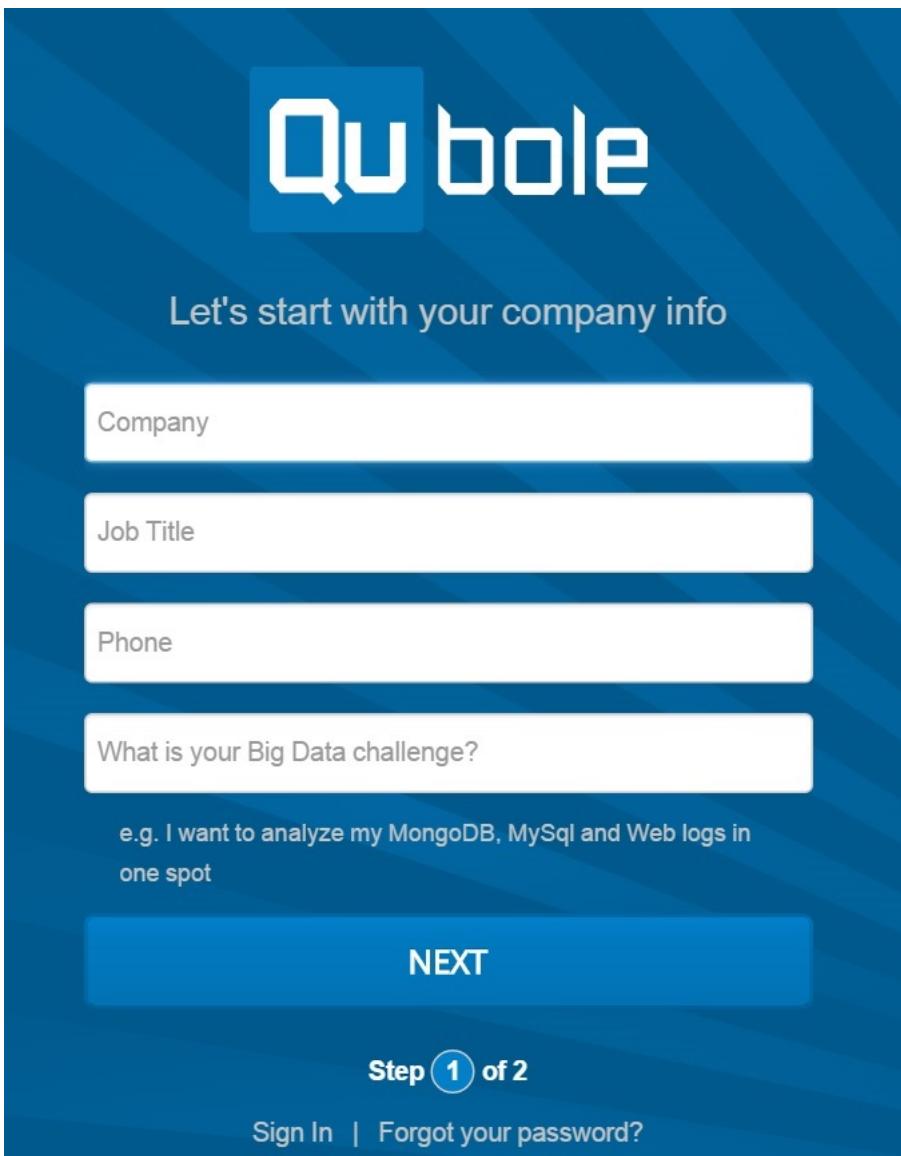
1.11 Azure Quick Start Guide

This document is intended for new users to quickly start up on Qubole Data Services by running simple Hadoop Job.

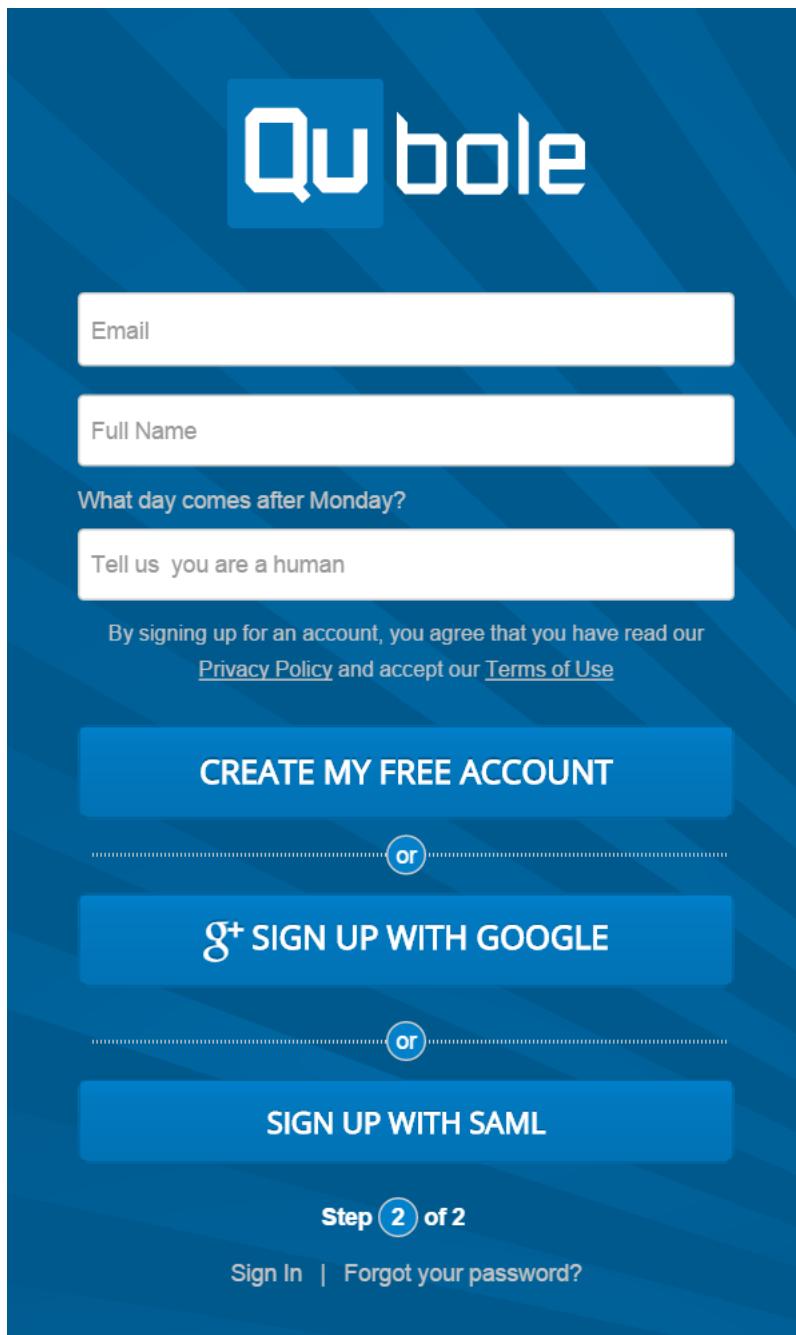
1.11.1 Getting Started with Qubole on Azure

How to Sign up

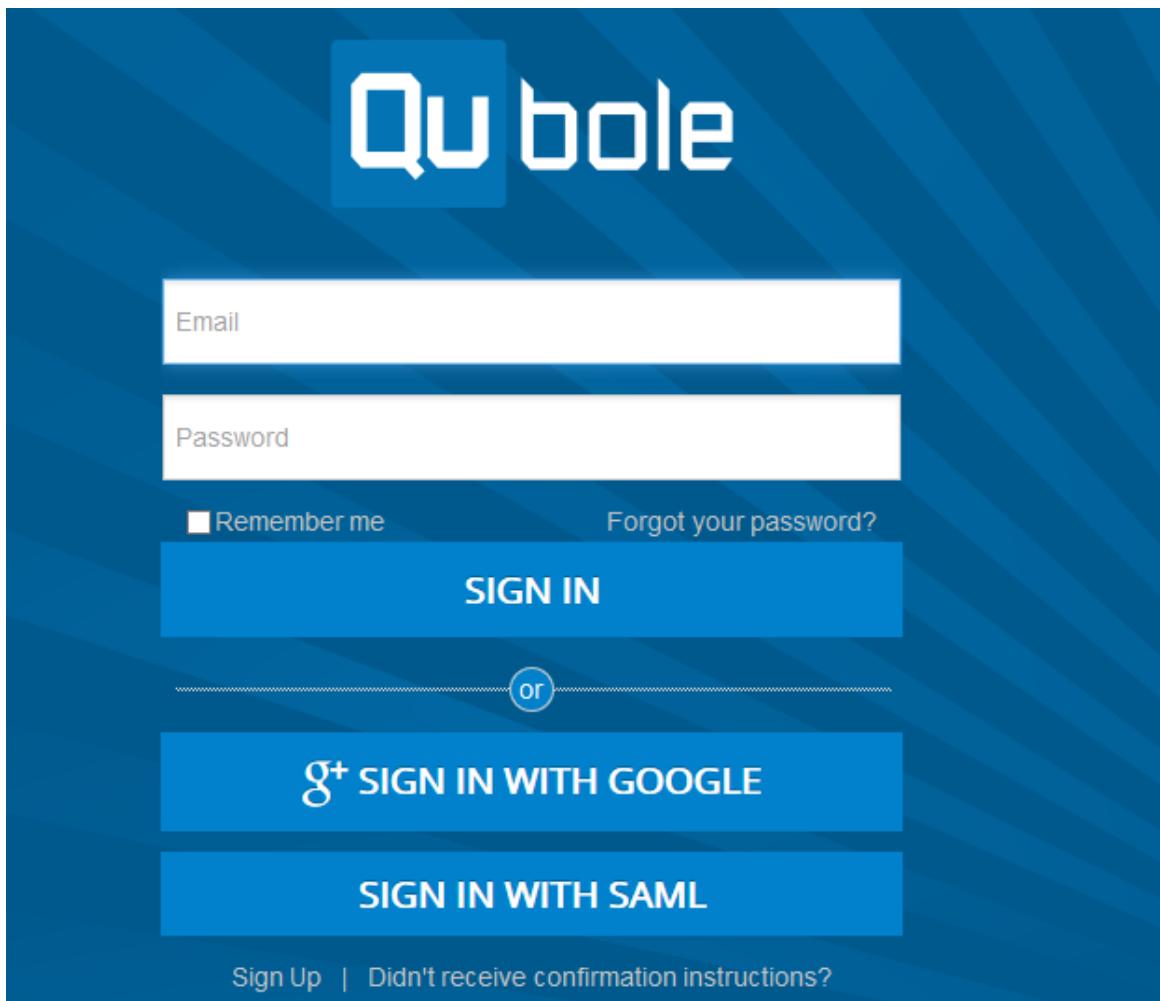
1. Go to <http://azure.qubole.com/>.
2. Click **Sign Up**. The following screen is displayed.



3. Provide the required information and click **Next**. The following screen is displayed.



4. Enter your Email ID and Full Name. Click **CREATE MY FREE ACCOUNT**. You will receive an email, to the email ID that you provided, with an activation code. You can choose to confirm your account by clicking on the link sent to you in the email or copy and paste the activation code in the signup window. Sign up and you will be provided with a free trial account.
5. After sign up, you can use the following login page to login into Qubole Data Services.



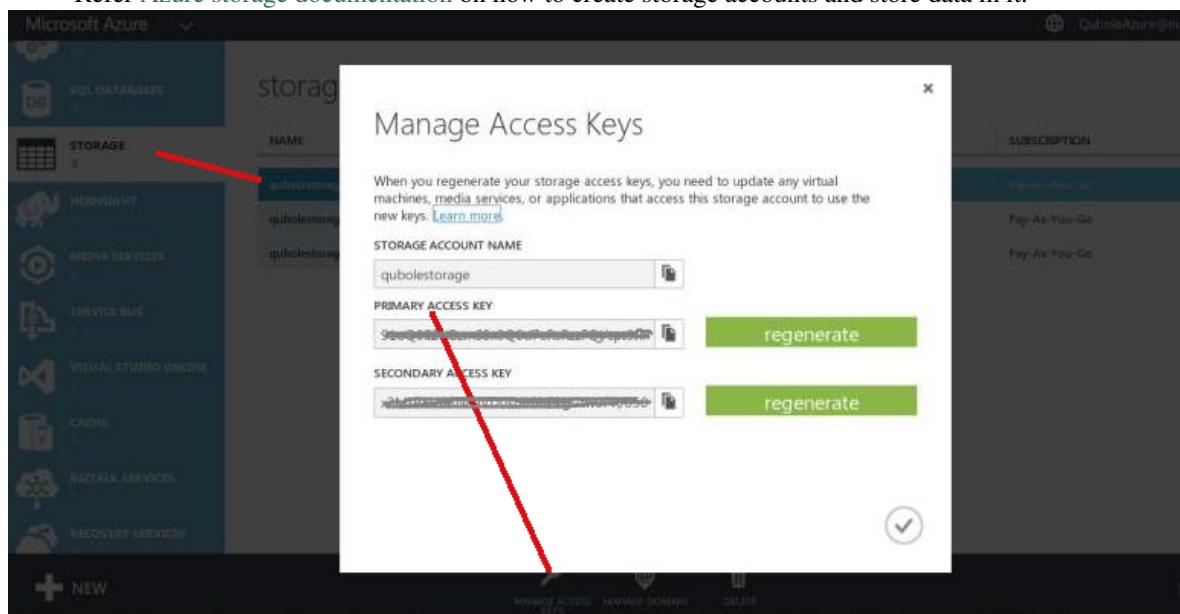
6. Once you login, you will see the **Analyze Pane**.

The image shows the Qubole Analyze pane. The left sidebar has icons for History, Composer, Data Wizard, Hive Bootstrap, Sessions, Analyze (selected), Templates, Schedule, Control Panel, and Tutorials. The main area has tabs for History, Composer, Data Wizard, Hive Bootstrap, and Sessions. A search bar at the top contains the text 'qbol_user_id>All'. Below the search bar is a table with columns 'Command', 'Type', and 'Time'. A message in the table says 'No commands found. Try with 'All' users or broaden your search criteria'. To the right of the table are sections for 'Sample Results' and 'Log'.

7. Click on the Control Panel on the left pane and provide the storage credentials.

- Default Location:** Location where all the logs, output, etc should be stored in, e.g `data@quboletedastore.blob.core.windows.net/defloc`
- Azure Storage Account:** Name of azure storage account where the data resides.
- Azure Storage Container:** Container in the specified storage account where the data resides.
- Azure Storage Access keys:** Key to access the storage account.

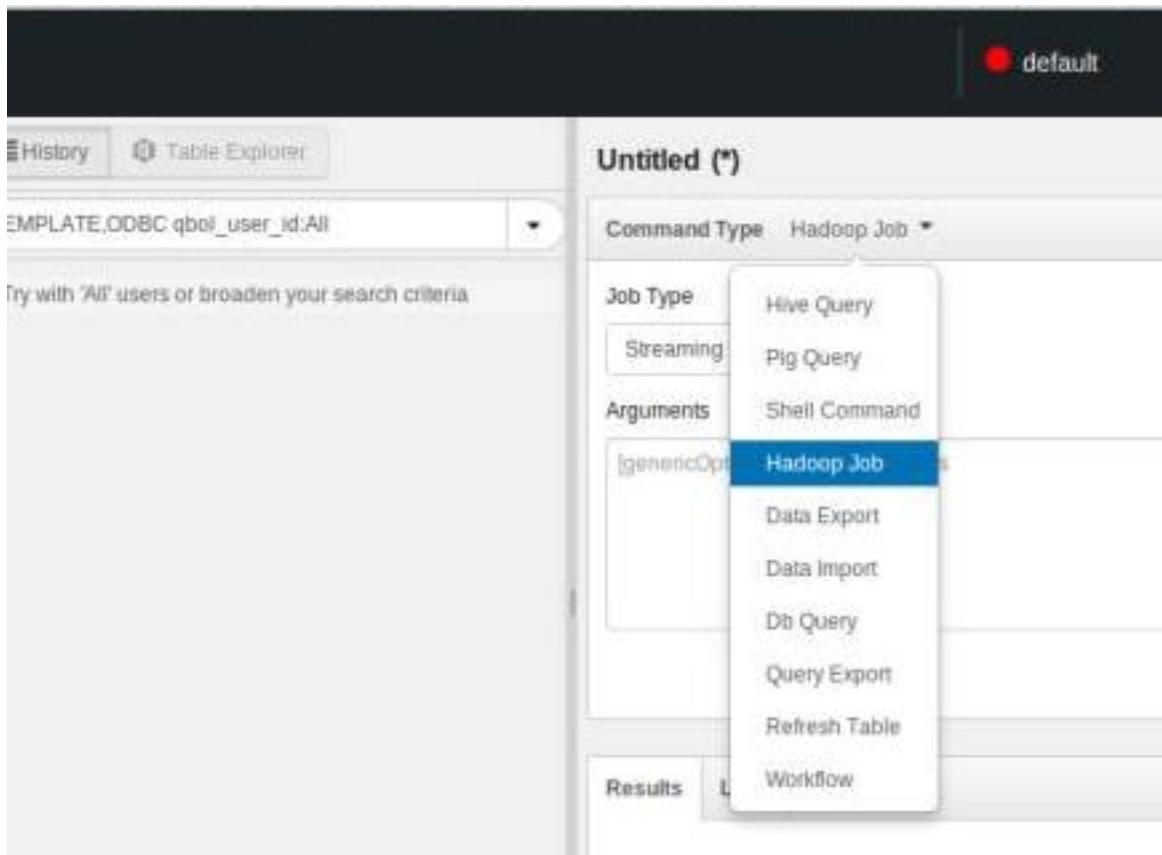
Refer [Azure storage documentation](#) on how to create storage accounts and store data in it.



8. The **Analyze Pane** consists of several tabs and editor window that help you to perform various tasks.
9. The **History** tab lets you retrieve historical command information.
10. The **Compose** button when clicked opens a command editor that lets you perform various tasks like running Hive queries, Hadoop streaming jobs, creating required commands, importing and exporting data, and so on. The Table Explorer tab will list all the tables that are available, once created.
11. The **Data Wizard** tab lets you connect to an external repository, view data, and import necessary data.
12. The **Hive Bootstrap** tab lets you configure the Hive settings for all your Hive queries.
13. The **Sessions** tab lets you create new sessions, run hive commands, and control auto scaling in the Hadoop cluster.

How to run your first Hadoop Job, simple word count streaming job

1. Ensure you have some sample file available in the storage account/container specified earlier in the storage settings tab.
2. In the **Analyze Pane**, click the **Compose** button. You will see a drop-down list, just below the tabs, with a list of tasks that you can perform. By default, **Hive Query** is selected or displayed, select **Hadoop Job**.



3. To run a Hadoop streaming job, ensure that the **Streaming** is selected under job type. In the **Compose** window, type a simple query. For example:

```
-mapper wc -numReduceTasks 0 -input asv://container@storageaccount.blob.core.windows.net/sample.
```

The storage credentials (storage account name, storage account access key) are used to access the input file and output folder.

You can also use some sample files available in the qubole's public container (`asv://data@qubolestorage.blob.core.windows.net/sample_files/file.txt`)

4. The word count output is now available on the output location specified in the above hadoop job.
5. This will bring up a hadoop cluster, you can see the cluster details under the **Control Panel** in the left panel.
6. To execute another query, click **Clear**. This clears the command window.
7. Now type and execute any other query in the **Composer** window.

Congratulations! You have just executed your first Hadoop query on the Qubole Data Service. Drop an email to help@qubole.com and one of our engineers will get back to you and help you onboard.

1.11.2 Enable Qubole to bring up clusters in your Azure Subscription

Enter details of your Azure Subscription

1. Click on the **Control Panel** on the left panel and go to the Account Settings. Change the compute type under Compute Settings at the bottom of the page to **CUSTOMER_MANAGED**.

Azure storage container

Azure storage access key

Save Reset

Compute Settings

Compute type **CUSTOMER_MANAGED**

Save Reset

2. Please edit the cluster settings to add the azure subscription details.

Account Settings

Cluster

Hive Bootstrap

Sessions

Manage Users

Subscription & Payment

My Profile

My Accounts

Active Cluster(s) Deleted Cluster(s)

Search : Enter Search Text

ID	Labels	Nodes	Up Time	Resources	Action
18 ✓	default	0			Start Edit

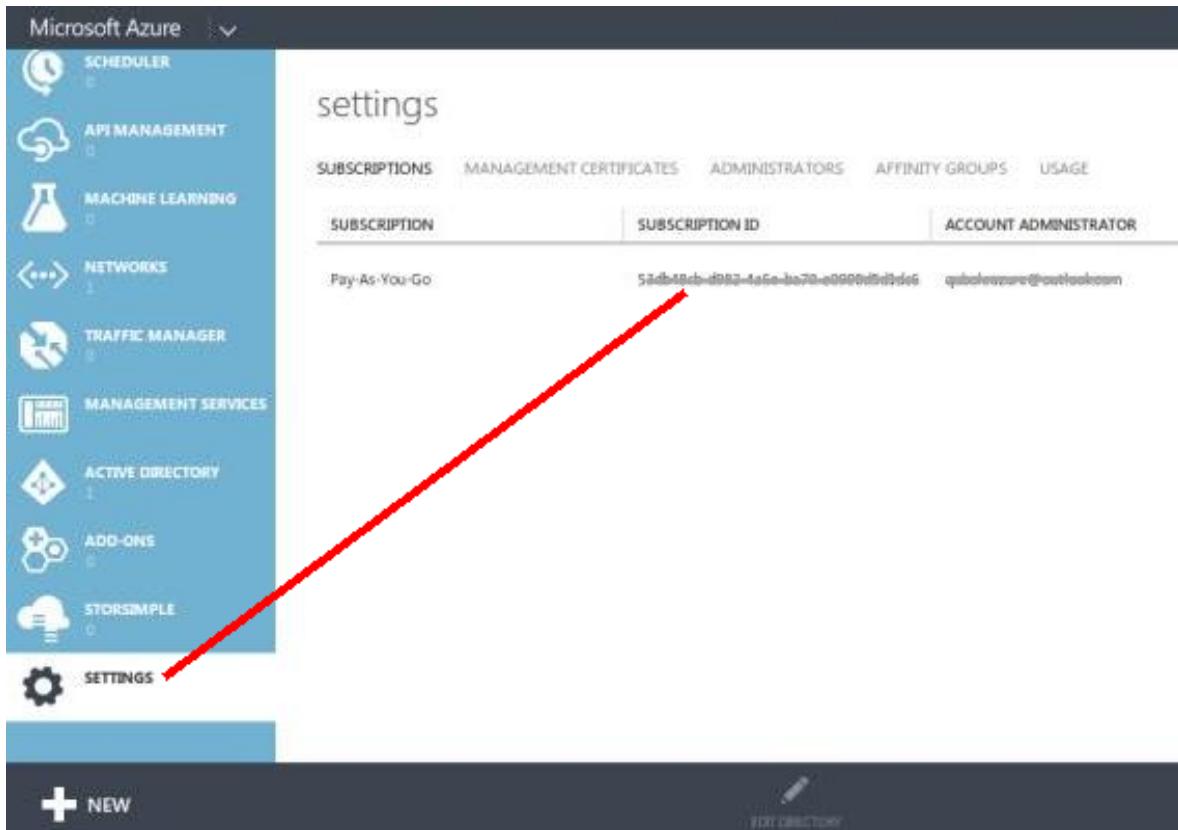
Click to edit cluster details

3. Enter the details here.

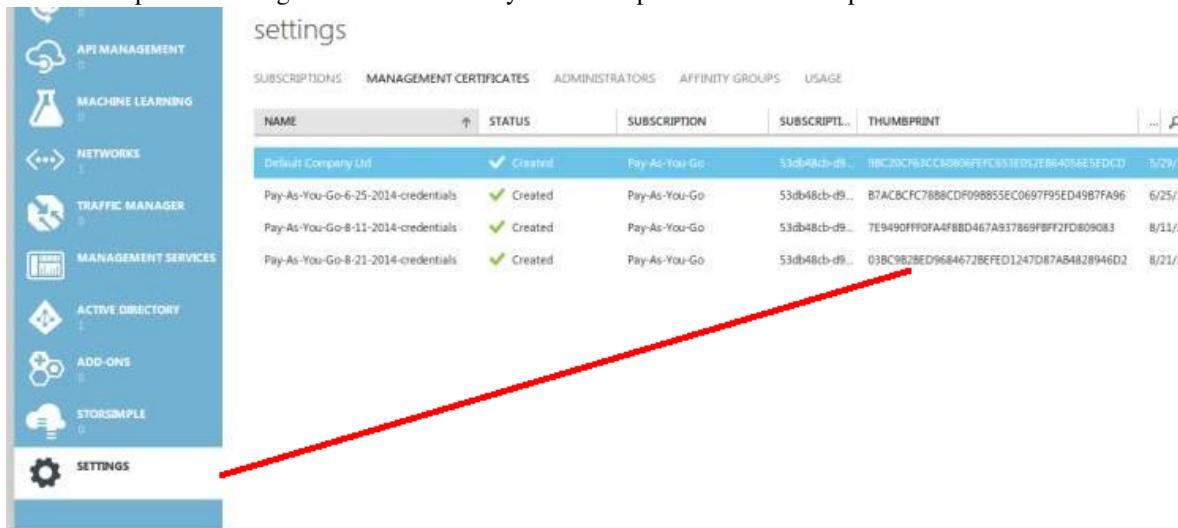
The screenshot shows the Qubole Data Service web interface. The left sidebar contains navigation links: Overview, Explore, SmartQuery, Analyze, Control Panel, and Tutorials. The main content area has a sidebar titled "Cluster" with options: Account Settings, Cluster, Hive Bootstrap, Sessions, Manage Users, Subscription & Payment, My Profile, and My Accounts. The "Cluster" option is selected. The main panel displays "Azure Settings" with fields for Azure Subscription Id, Azure Certificate Thumbprint, Azure Service Certificate, Azure Compute Region (set to East Asia), Azure Network Name, Azure Affinity Group, Azure Storage Account, Azure Atorage Container, and Azure Storage Access Key. Below this is a section titled "Hadoop Cluster Settings".

Use the following instructions to get these details from your azure subscription.

Azure Subscription Id: Present under the settings tab of the Azure Management Portal



Azure Certificate Thumbprint: Click on <https://manage.windowsazure.com/publishsettings>. This will prompt you for the login credentials (if not logged in already), a publish_settings file will get downloaded. This will also upload a management certificate on your subscription. Use the thumbprint of this certificate.



Azure Service Certificate: The service certificate needs to be extracted from the publish_settings file, that got downloaded in the previous step. Enter the base 64 encoded content between the quotes after ManagementCertificate as the service certificate.

```
<?xml version="1.0" encoding="utf-8"?>
<PublishData>
  <PublishProfile>
```

```

PublishMethod="AzureServiceManagementAPI"
Url="https://management.core.windows.net"
ManagementCertificate="XXXXXXXXXXXXXX"
<Subscription
  Id="53db48cb-d982-4a6e-ba70-e0998d5d1dc6"
  Name="Pay-As-You-Go" />
</PublishProfile>
</PublishData>

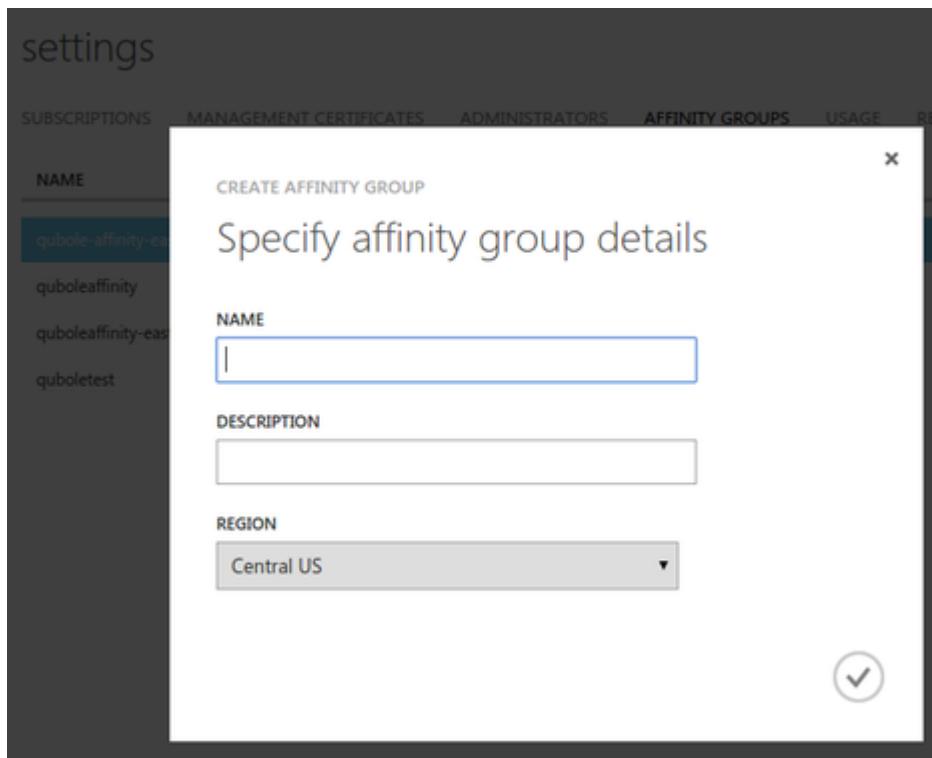
```

Refer [Azure documentation](#) for more details (Note: The document is a little outdated).

Azure Compute Region: Select your preferred region.

Azure Network Name: Create a network in your **preferred region** and provide the name here. Refer [Azure network documentation](#) if you need help in creating network.

Azure Affinity Group: Create an affinity group in your **preferred region** and provide the name here. Navigate to the **Settings** tab > **Affinity Groups**. Create an affinity group. The **Create Affinity Group** dialog is as shown in the following figure.



Refer [Azure documentation](#) if you need help in creating network.

Azure Storage Account: Create a locally redundant storage account in the **above affinity group** and provide the name here. The storage account will contain the OS HDs of the Virtual Machines that come up in your hadoop cluster.

Azure Storage Container: Create a container in the above **storage account** and provide the name here.

Azure Storage Access Key: Access key of the storage account provided.

1.11.3 Azure Subscription and Service Limits, Quotas, and Constraints

Refer to the [Azure documentation](#) to check the Microsoft Azure limits. If you want to run larger clusters, please contact Microsoft support to get the quotas increased.

Example Configuration:

Cluster Size	Machine Type	Cores	Cloud Services
49 slaves + 1 master	Small - 1 vcpu, 1.75GB mem	50	50
49 slaves + 1 master	Medium - 2 vcpu, 3.5GB mem	100	50
49 slaves + 1 master	Large - 4 vcpu, 7GB mem	200	50

No of cores = No of CPUs per machine * Cluster size

No of cloud services = Cluster Size

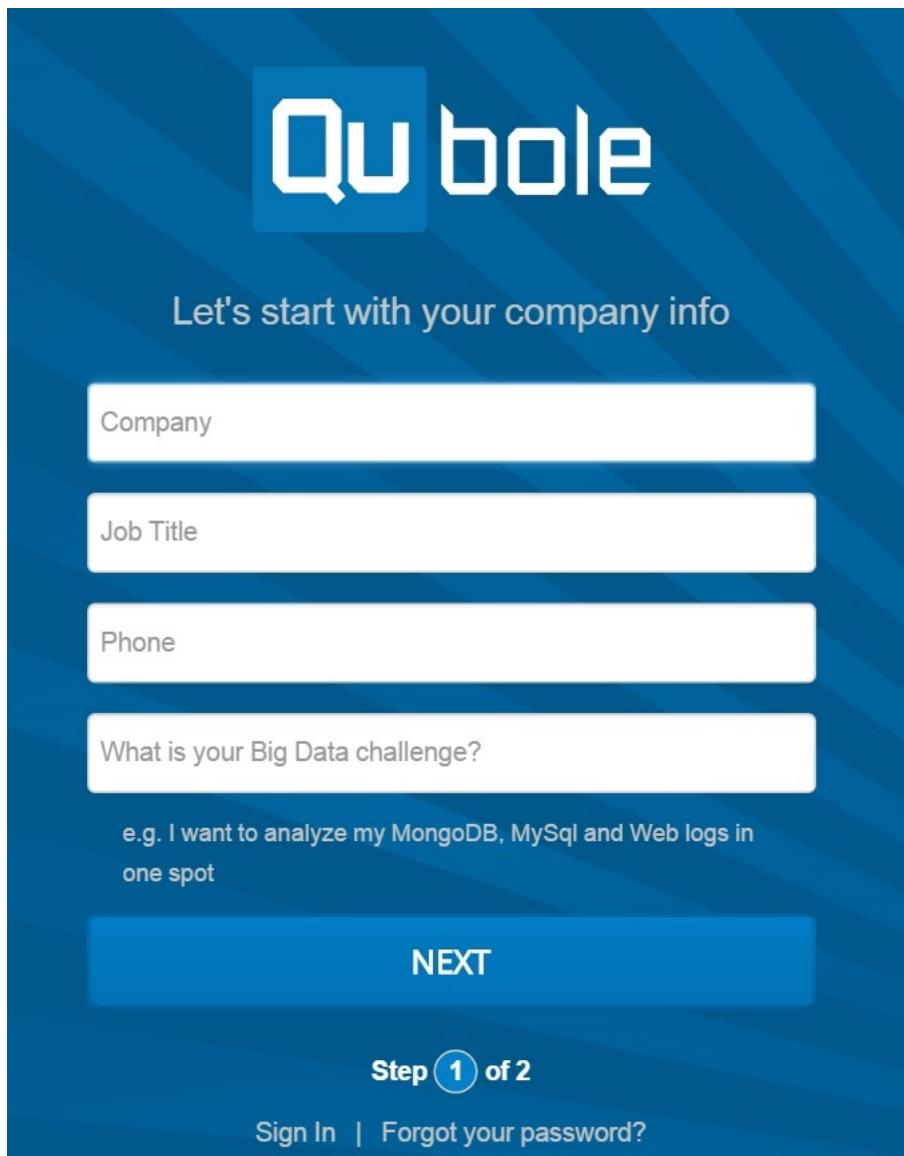
1.12 GCE Quick Start Guide

This document is intended for new users to quickly start up on Qubole Data Services by running simple Hadoop Job.

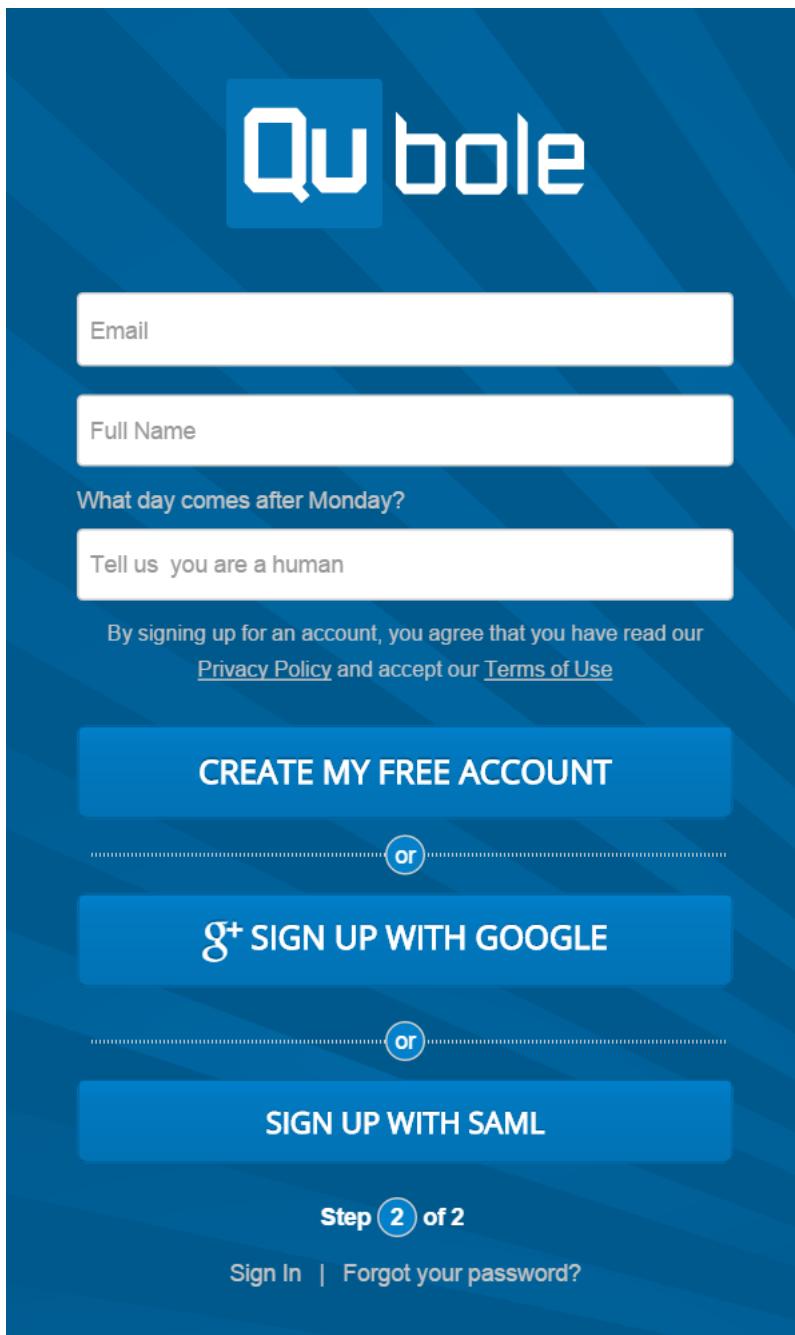
1.12.1 Getting Started with Qubole on GCE

How to Sign up

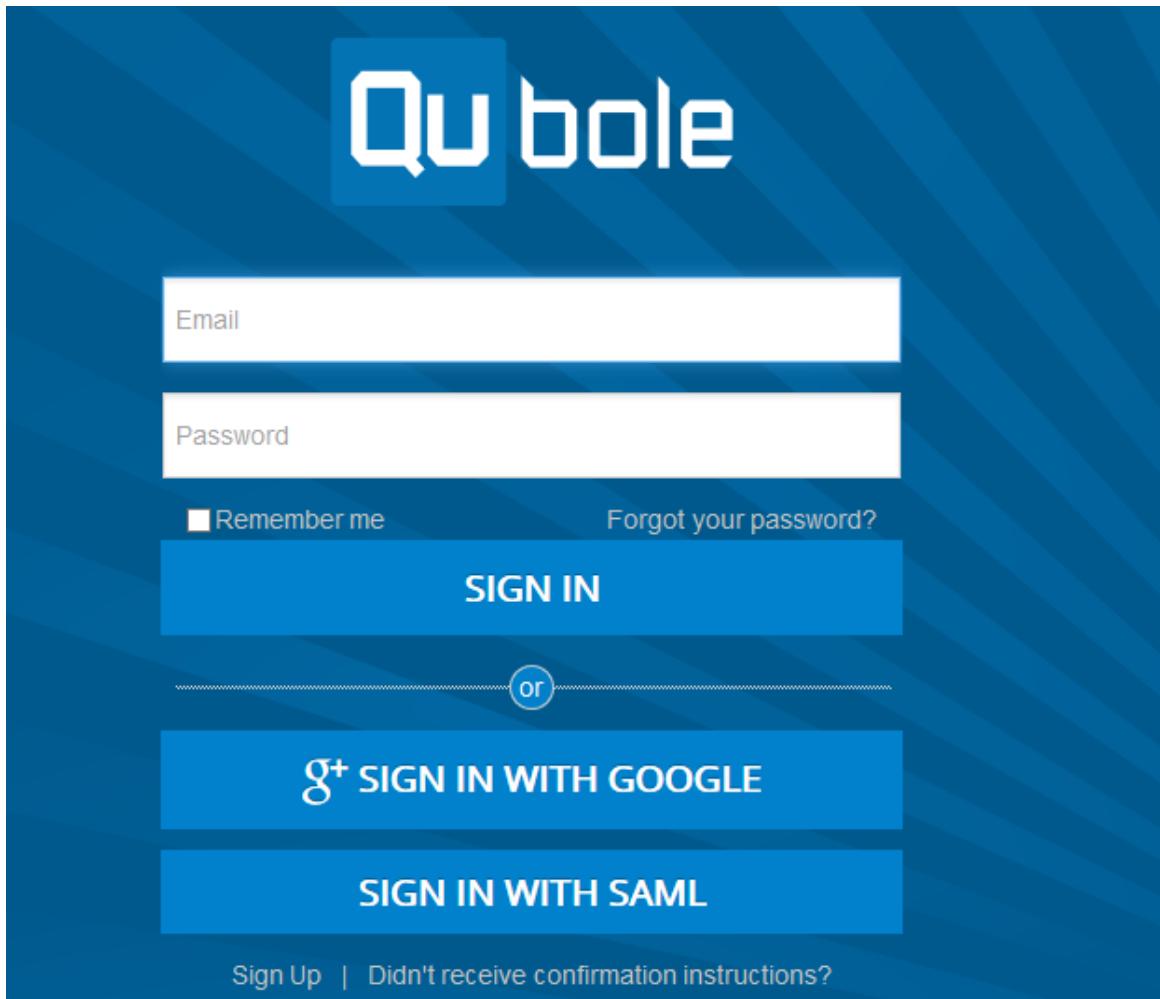
1. Go to <http://gce.qubole.com/>.
2. Click **Sign Up**. The following screen is displayed.



3. Provide the required information and click **Next**. The following screen is displayed.



4. Enter your Email ID and Full Name. Click **CREATE MY FREE ACCOUNT**. You will receive an email, to the email ID that you provided, with an activation code. You can choose to confirm your account by clicking on the link sent to you in the email or copy and paste the activation code in the signup window. Sign up and you are provided with a free trial account.
5. After signing up, you can use the following login page to login into Qubole Data Service.



6. Once you login, you see the **Analyze** page.

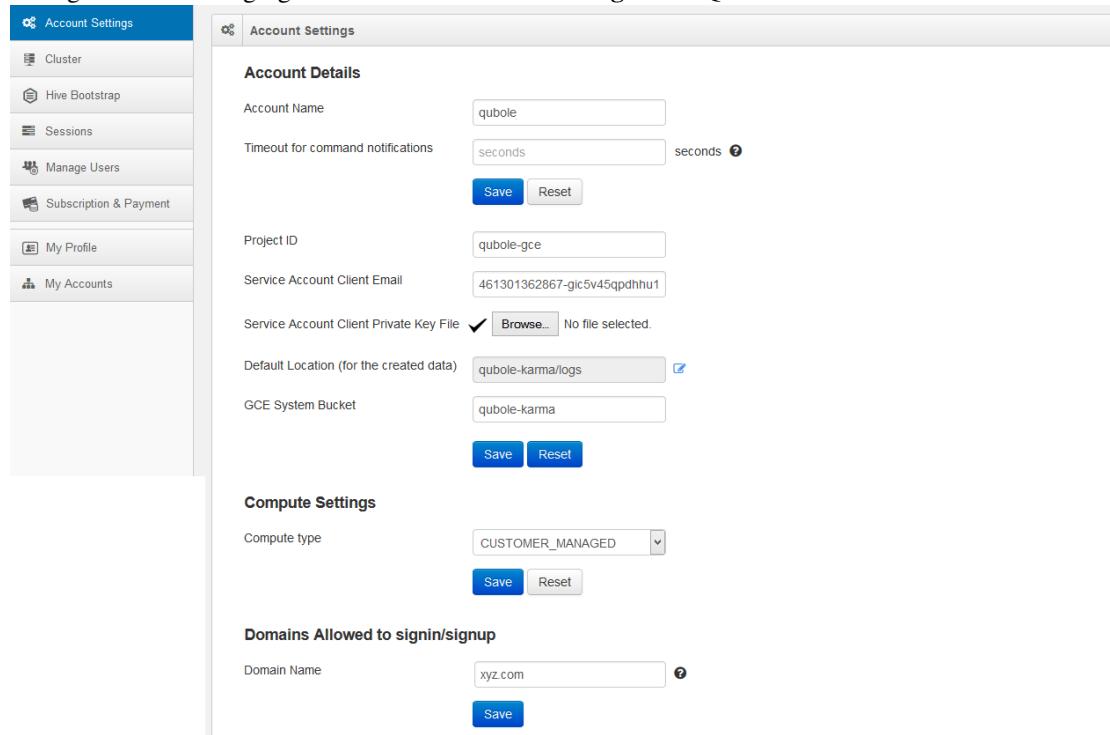
 A screenshot of the Qubole Analyze page. The top navigation bar includes 'Compose', 'History', and 'Tables' tabs. The main search bar contains the query 'interface:API,UI,TEMPLATE,ODBC,SMART_QUERY qbol_user_id>All'. Below the search bar is a list of recent queries:

- 15680 ☆ select user_id from sent_emails; Hive Query 07/22 09:27
- 15679 ☆ show tables; Hive Query 07/22 09:26
- 15282 ☆ select user_id from sent_emails limit 1; Hive Query 06/19 17:17
- 14950 ☆ select user_id from sent_emails limit 1; Hive Query 04/22 10:30
- 14949 ☆ select user_id from sent_emails; Hive Query 04/22 10:26
- ps

Configuring Qubole Account Settings

Prerequisites: You must have an account on GCE cloud to configure a Qubole account.

Navigate to [Control Panel](#) to see the **Clusters** tab by default. Click **Account Settings** to configure account and compute settings. The following figure shows the **Account Settings** of an Qubole account on GCE.



The options in the Account settings are as follows:

- **Account Name** - Set a name to the account.
- **Timeout for Command notification** - It is set in seconds for the queries that you run in the **Compose** tab of the [Analyze](#) page.
- **Project ID** - Enter the project ID in this text field. The project ID is displayed in the GCE account page as shown the following figure.

Project: qubole-gce

Service accounts

Email address ^

461301362867-1et9k293bupobupobupobupobupop@developer.gserviceaccount.com

Email address is listed in **Service accounts**. Copy the email address and enter it in the **Service Account Client Email** text field.

- **Service Account Client Private Key File** - In the figure provided above, click the client email address that is entered as the service account client email. **Service account** details are displayed as shown in the following figure.

Service account

Client ID	461301362867-pr4i7mnkmnnkmmnkmmnkmmnkmmnkms.googleusercontent.com
Email address	461301362867-pr4i7mnkmnnkmmnkmmnkmmnkmmnk@developer.gserviceaccount.com
Certificate fingerprints	55f2855f28d55f2855f2855f2855f2855f255f28

Done

Click **Generate new P12 Key** to get the private key that is downloaded to a default location. Browse to the location that contains the private key.

- **Default Location (for the created data)** - It is a location where logs and output data, and so on are stored. Change it if you want a different default location.
- **GCE System Bucket** - Enter the system bucket in this text field.
- **Compute Settings** - By default, the **Compute Type** displayed is **QUBOLE_MANAGED**. The other option in the drop-down list is **CUSTOMER_MANAGED**. **QUBOLE_MANAGED** is available for a 15-day trial period. After the trial expires, you must set it to **CUSTOMER MANAGED**.
- **Domains Allowed to signin/signup** - You can select the domains that you want to sign up and sign in. You can set multiple domains separated by a comma. Specify the domain in the **Domain Name** field that is available in **Domains Allowed to signin/signup**.

After specifying domains, click **Save**.

Running a Simple Word Count Hadoop Streaming Job

Perform the following steps to run a Hadoop streaming job:

1. Navigate to the [Analyze](#) page from the top menu and click the **Compose** button.
2. Clicking **Compose** opens a command editor. Select the command type as **Hadoop Job** from the drop-down list.
3. Specify the location of the job JAR file (in this case: `gs://qubole-karma/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar`)
4. Specify the arguments to the JAR file. In the example shown below, specify the mapper/reducer scripts, the location of these scripts, the number of reducers and the location of the input dataset, and an output GCE bucket location as arguments that are as shown below.

```
-mapper wc -numReduceTasks 1
-input gs://qubole-karma/default-datasets/gutenberg
-output gs://.../...
```

Note: The output path shown in the above step and the following figure is not an actual path. Provide an output location in an google storage bucket that you own.

A sample Hadoop job is shown in the following figure.

The screenshot shows the Qubole Query Composer interface with the following configuration:

- Command Type:** Hadoop Job
- Job Type:** Custom JAR
- Path to JAR File:** gs://qubole-karma/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar
- Arguments:**

```
-mapper wc -numReduceTasks 1
-input gs://qubole-karma/default-datasets/gutenberg
-output gs://.../...
```

- Click **Run** to execute the job. The status of the job is displayed on the top of the query composer. The query result is displayed in the **Results** tab.

Congratulations! You have executed a first Hadoop command using Qubole Data Service.

Enable Qubole Clusters for GCE

The account settings are as described in [Configuring Qubole Account Settings](#). To use Qubole on GCE, you must configure storage settings.

Qubole supports Hadoop clusters. Add a Hadoop clusters for GCE in the **Control Panel**. Navigate to [Control Panel](#) to see the **Clusters** tab by default as shown in the following figure.

The screenshot shows the Qubole Control Panel Clusters tab with the following data:

Active Cluster(s)		Deleted Cluster(s)					
+ Add Cluster		Edit Cluster					
Search : <input type="text" value="Enter Search Text"/> X							
Id	Labels	Nodes	Up Time	Resources	Action		
33	default	0			Edit	Delete	More Options

Click the + icon to add a new cluster and edit icon to edit an existing cluster.

The cluster configuration page is as shown in the following figure.

Cluster Settings

Cluster Labels

Cluster Type

GCE Settings

Project ID

Service Account Client Email

Storage system bucket

Region

Service Account Client Private Key File No file selected.

Hadoop Cluster Settings

Minimum Slave Count

Maximum Slave Count

Master Node Type

Slave Node Type

Node bootstrap file

Override Hadoop Configuration Variables
 Info: if you are configuring fair scheduler, make sure to add fairSharePreemptionTimeout property.
 Example: <fairSharePreemptionTimeout></fairSharePreemptionTimeout>

Fair Scheduler Configuration

Default Fair Scheduler Pool

Other Settings Disable Automatic Cluster Termination

In **Cluster Settings**, add a label in the **Cluster Labels** text field. This is a mandatory step. Select **Hadoop 2** or **Spark** as the cluster type if you do not want the default **Hadoop** cluster type.

In **GCE Settings**, the options to set are:

- **Project ID:** Specify the project ID. [Configuring Qubole Account Settings](#) describes how to find it.
- **Service Account Client Email:** Specify the service account client email. [Configuring Qubole Account Settings](#) describes how to get it.
- **Storage system bucket:** Specify a GCE storage system bucket.

- **Region:** Select a different region if you do not want the default region.
- **Service Account Client Private Key File:** Browse to the key file. *Configuring Qubole Account Settings* describes how to download it.

In **Hadoop Cluster Settings**, the options to set are:

- **Minimum Slave Count:** Set the minimum number of slave nodes if you want to change it from the default 2.
- **Maximum Slave Count:** Set the maximum number of slave nodes if you want to change it from the default 2.
- **Master Node Type:** Set the master node type by selecting the preferred node type from the drop-down list.
- **Slave Node Type:** Set the slave node type by selecting the preferred node type from the drop-down list.
- **Node Bootstrap File:** Set the node bootstrap file.
- Override the Hadoop configuration by entering the variables in the **Override Hadoop Configuration Variables** text field.
- Set the new **Fair Scheduler Configuration** values to override the default values.
- Specify the **Default Fair Scheduler Pool** if the pool is not submitted during job submission.
- **Other Settings:** To enable/disable **Disable Automatic Cluster Termination**.

Click **Save** to add a Hadoop cluster. Click **Cancel** to not add a cluster.

GCE Machine Types

For more information on the machine types, see [machine types](#).

Example Configuration:

Machine name	Cores	Memory
n1-standard-1	1	3.75GB
n1-standard-2	2	7.5GB
n1-standard-4	4	15GB
n1-standard-8	8	30GB
n1-standard-16	16	60GB
n1-highmem-2	2	13GB
n1-highmem-4	4	26GB
n1-highmem-8	8	52GB
n1-highmem-16	16	104GB
n1-highcpu-2	2	1.8GB
n1-highcpu-4	4	3.6GB
n1-highcpu-8	8	7.2GB
n1-highcpu-16	16	14.4GB
f1-micro	1	0.6GB
g1-small	1	1.7GB

Qubole User Guide

2.1 Introduction

Qubole Data Service (QDS) provides a highly integrated set of tools that help organizations to analyze data and build reliable data driven applications in the cloud. QDS takes care of orchestrating and managing cloud resources. Hence facilitates the users to focus on analyzing the data and applying their domain expertise to creatively use the data.

QDS provides the following services:

Cloud Data Warehouse: QDS provides Apache Hive as a service in the Cloud. This allows the user to define schemas over data sets stored in the Cloud Storage systems, like AWS S3.

Analyze: This is the interface that provides on-demand computational abilities over historical data in S3. This allows the users to run Hive queries and/or Map-Reduce jobs as commands in QPal to analyze historical data in S3. This also allows the users to run commands to import and export data sets from external systems.

Scheduler: This is a service to run recurring commands that includes sophisticated work flows. For example, the build reporting applications.

All these services are hosted in Amazon AWS cloud and are exposed through a browser based application as well as REST APIs.

2.2 Features

2.2.1 Users and Accounts

Users can [Signup](#) for Qubole using an email address – either by supplying one directly – or by authorizing Qubole to access basic profile information (including email) from providers like Google. In either case – users are identified internally within Qubole using email addresses – and where users must refer to another user – they must do so using the email address of the other user.

Each user can be a member of one more accounts. A user who does not belong to any account cannot perform any meaningful actions. All non-trivial user actions (like creating a table, running a command, importing or exporting data or even looking up older commands etc.) must be made in the context of an account. Here are some salient traits of an account:

1. Any user can create a *free* account by going to the [Create Account](#) page. Qubole may put reasonable restrictions on the number of free accounts that a user can be associated with.

2. While creating an account – user has to provide credentials that will allow Qubole to orchestrate AWS resources on behalf of the account like AWS credentials to read S3 buckets (Storage Credentials) and to spawn machines in EC2 (Compute Credentials).
3. Every account has one or more *admin* users. The user who created the account is an admin by default.
4. Admin users can invite more users by sending invites to their email addresses. Similarly, they can remove users or change the admin status of users from the [User Management](#) page.
5. Every account has a distinct Hive warehouse as well as a common (on-demand) Hadoop cluster – that is used for running queries from member users and background jobs.

When accessing Qubole via the Browser – users with multiple accounts are always shown the current account on the right corner and they can also switch between accounts by the controls placed therein. For API access using tokens, a user must get and use tokens specific to a particular account. Tokens can be obtained/reset from the [My Accounts](#) tab in the **Control Panel** page from where they can also administer other aspects of their associated accounts (selecting a default account for example).

2.2.2 Explore

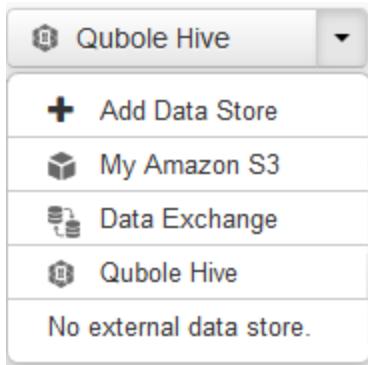
Using QDS in Data Exploration

Data Exploration helps in making critical decisions to do better data analysis. It helps in visualising and understanding big data to improve decision making.

Use QDS [Explore](#) to search data from various sources in addition to By default, you see the Qubole Hive database with the expanded default database as shown in the following figure.

The screenshot shows the Qubole Hive interface. On the left, there is a sidebar with a dropdown menu set to "Qubole Hive". Below it is a search bar with "Search in default" and a "Go!" button. A refresh icon is also present. The main area is titled "Qubole Hive" and contains two tabs: "Properties" and "Access Details". Under "Access Details", the "Number of Schema(s)" is listed as 4. Below this, there is a list of schemas: "3int" and "3int_0".

For accessing **My Amazon S3** and adding an external data store, pull the drop-down list as shown in the following figure.



Only a system administrator can add a data store and hence, see the Add Data Store option.

QDS Explore supports managing big data in:

- Gaining faster time to insight and understanding the wider range of big data assets
- Improving analysis productivity due to accuracy
- Viewing the data in Hive tables available in the Hive metastore
- Connecting to any supported databases and viewing sample data from its tables
- Connecting to S3 buckets and viewing the sample data
- Sharing hive table data stored in S3 locations with partners or third-party users. See [Data Exchange](#) for more information.

The following sections explain the functions of the Explore feature:

- [Exploring Data in Qubole Hive](#)
- [Analyzing Data in Hive Tables](#)
- [Exporting Data from the Hive Metastore](#)
- [Adding a Data Store](#)
- [Exploring Data in Amazon S3](#)
- [Exporting Data from Amazon S3 Tables](#)
- [Creating Schema from Amazon S3 Data](#)
- [Adding Data Exchange Spaces](#)
- [Managing My Spaces](#)
- [Managing Other's Spaces](#)

Exploring Data in Qubole Hive

Qubole Hive metastore is a database that contains data created in hive tables. Data in the Qubole Hive metastore can be explored before analyzing. You can create databases and tables using a Hive command.

See [Hive in Qubole](#) and [Submit a Hive Command](#) for more information.

By default, **Explore** displays the Qubole Hive metastore. Clicking a table displays table properties. Click **Rows** to see the sample data. A hive table with properties and sample data is shown in the following figure.

Qubole Hive > default_qubole_memetracker

Rows	Properties			
Table Properties				
Name: default_qubole_memetracker				
Column Properties				
Column	Data Type	Comments		
site	string			
ts	string			
phr	string			
lnks	string			
month	string			

Qubole Hive > default_qubole_memetracker

Rows	Properties			
site				
http://codeproject.com/kb/silverlight/convertsilverlightcontrol.aspx	ts	["how to create prc		
http://wallstreetexaminer.com/?p=2987		[]		
http://news.bbc.co.uk/go/rss/-/1/hi/scotland/highlands_and_islands/7535558.stm		["our continuing sti		
http://news.bbc.co.uk/go/rss/-/1/hi/scotland/south_of_scotland/7535392.stm		["safeguard our fis		
http://news.bbc.co.uk/go/rss/-/1/hi/scotland/north_east/7535090.stm		["month to month",		
http://ww1.rtp.pt/noticias/index.php?article=357117&visual=26&rss=0		["inclus o social", "		
http://marinelog.com/docs/newsmmvii/2008jul00312.html		["it is no longer fea		
http://maitland.yourguide.com.au/news/local/news/general/foul-odour-unmasked/1232399.aspx?src=rss		["for four weeks be		
http://mourningserial.wordpress.com/2008/07/26/beeswax_11		["no arguments no		
http://mourningserial.wordpress.com/2008/07/12/dartagnan_mourningserial_09		["still gets to ya dc		



Click the **full view icon**  to see the table in a full window.



Click the **refresh icon**  to refresh the table.



To analyze/export data from Hive tables or refresh tables, click the icon . A drop-down list is displayed as shown in the following figure.

Analyze Data
Refresh Table
Data Export
Export to Space ...

Select **Analyze Data** or **Refresh Table** or **Data Export** or **Export to Space** as required.

The **Export to Space** option is used in Data Exchange.

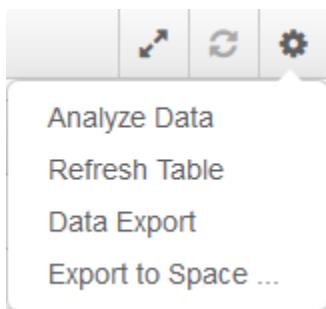
See [Analyzing Data in Hive Tables](#), [Exporting Data from the Hive Metastore](#), and [Data Exchange](#) for more information.

Qubole allows refreshing the hive schema to see new tables.

Analyzing Data in Hive Tables

From the Hive metastore, select the Hive table that requires data analysis. Click the icon .

A drop-down list is displayed as shown in the following figure.



Click **Analyze Data** from the list.

The query composer in [Analyze](#) opens in a new tab with the query.

Click **Run** to execute a query. The query result is displayed in the **Results** tab.

A sample hive query with a successful result is shown in the following figure.

a	b
43497754	37
43497755	1
43497756	22

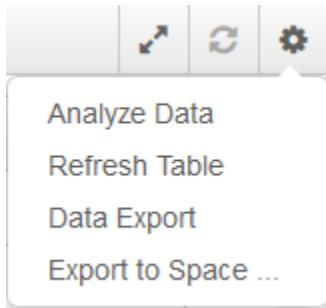
Exporting Data from the Hive Metastore

You can export data from the Hive metastore into an existing data store.



Select the Hive table and click the icon .

A drop-down list is displayed as shown in the following figure.



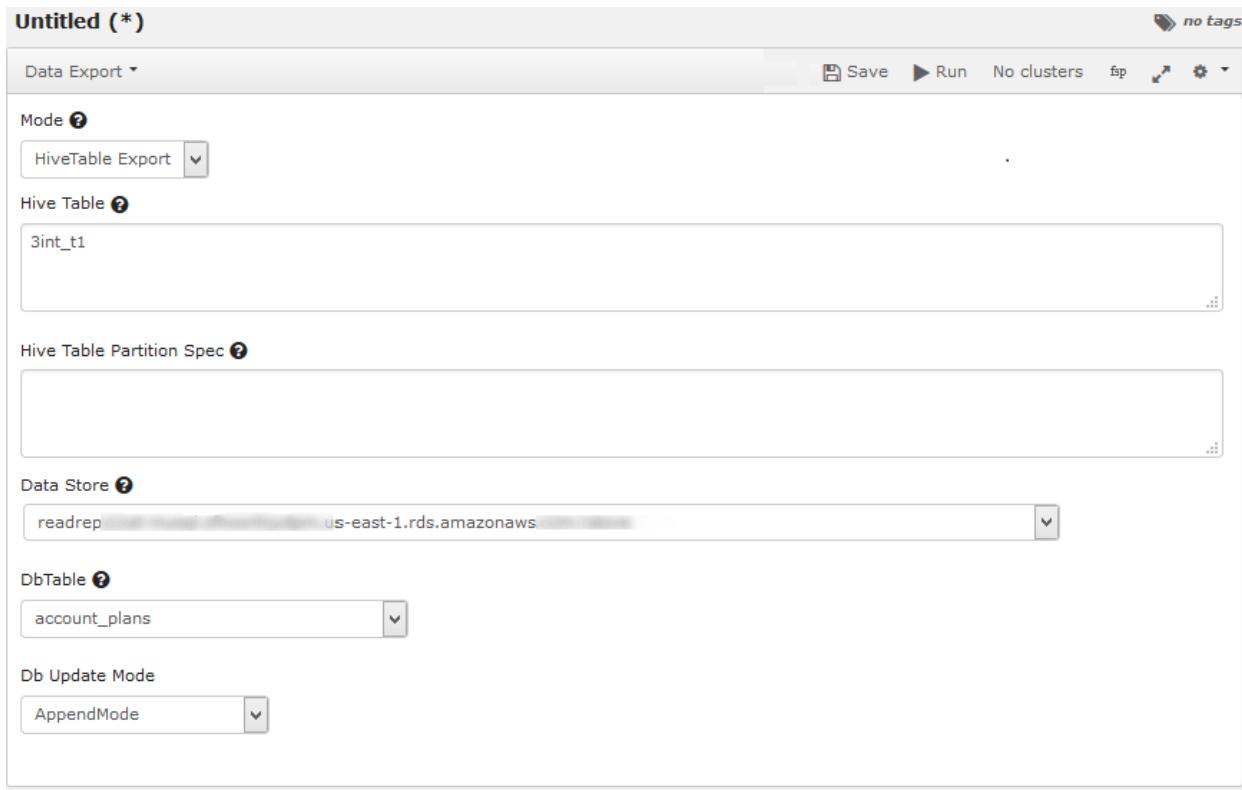
Click **Data Export** from the list.

The query composer of **Analyze** opens in another tab/window with **Command Type** as **Data Export** and **Mode** as **HiveTable Export**. The table name is mentioned in the **Hive Table** text field. There are other mandatory fields that must be filled before running the query.

Follow these steps to fill the required information before running a data export query:

1. Mention the partition in the **Hive Table Partition Spec** text field. Ensure that the name and value of partitions are in this format, **dt=20130101/country=US**. This is an optional step if you require partitions for a table.
2. From the **Data Store** drop-down list, select a data store.
3. After you select a data store, the tables get populated in the **DBTable** drop-down list. Select a database table from the list to which you want to export data.
4. In the **DB Update Mode** drop-down list, **Append mode** is selected by default. The other two options available in this list are **Update mode** and **Insert and update mode**. Currently, **Insert and update mode** is only available for MySQL data stores.

A sample hive table data export query is shown in the following figure.



- Click **Run** to execute a query. The query result is displayed in the **Results** tab.

Adding a Data Store

A data store is required to import/export data from/to an external relational database management system. You must be a system administrator to add a data store. Qubole Data Service (QDS) supports the following database types:

- Microsoft SQL Server
- Mongo DB
- MySQL
- Postgres
- Redshift
- Vertica

See [Data Store](#) for more information on data store and adding/creating a data store.

Exploring Data in Amazon S3

You can export data and create hive tables from the existing Amazon S3 data. The files containing data are maintained in folders, which are created in Amazon S3 buckets. You can upload a file to a folder on a S3 bucket and download a file from the folder on a S3 bucket. This setting can be enabled/disabled using the **Account Settings** in the [Control Panel](#) page. See [Managing Account Settings](#) for more information.

Select a file to see sample data. By default, **Properties** of the file are displayed. Click the **Sample Data** tab to see the data in text/JSON format. The following figure provides an example of **Sample Data** and **Properties** of a file.

The screenshot shows a data visualization tool with a header bar containing 'Sample Data' and 'Properties' tabs, and icons for refresh and settings. Below the header are buttons for 'Raw Data' (selected), 'Formatted', 'Format: Text' (dropdown), 'Delimiter: Tab' (dropdown), 'Create Schema', and 'Export Data'. The main area displays a grid of data with three columns: 'Col 0', 'Col 1', and 'Col 2'. The data rows are indexed from 0 to 10. The content of the grid is:

Col 0	Col 1	Col 2
0	32	70537
1	139	55718
2	15	12123
3	153	93703
4	135	76767
5	88	40610
6	99	23793
7	97	88964
8	137	13787
9	19	52565
10	137	99312

The screenshot shows a 'Properties' view for an S3 file. It includes tabs for 'Sample Data' and 'Properties'. The 'Properties' tab is active, displaying 'S3 File Properties' with two entries: 'File Path' set to 's3://test-qubole-s3-bucket/data1_30days/20100101/IN/1.tsv' and 'Region' set to 'us-east-1'.

By default, the data formatted in **Text** is displayed in the **Sample Data** tab. Select **JSON** from the drop-down list to see data in the JSON format. Click **Raw Data** to see the unformatted text.

The options available in the **Delimiter** drop-down list are:

- Comma (default)
- Tab
- Space
- Semicolon
- Ctrl-A
- Ctrl-B
- Ctrl-C

See [Exporting Data from Amazon S3 Tables](#) and [Creating Schema from Amazon S3 Data](#) for more information.

Exporting Data from Amazon S3 Tables

After analyzing data, you can export the data back to an external data store.

Follow these steps to export data from a S3 file to a data store:

1. Select the file from which you want to export data.

2. From the **Sample Data** tab, select the format and delimiter if you do not want the default format/delimiter. Click

the **Export Data** button. Alternatively, click the icon  to get a drop-down list as shown in the following figure.



Select **Export S3 Data**.

3. Once you click **Export Data**, the query composer of **Analyze** opens in another tab with **Command Type** as **Data Export** and **Mode** as **Directory Export** selected.

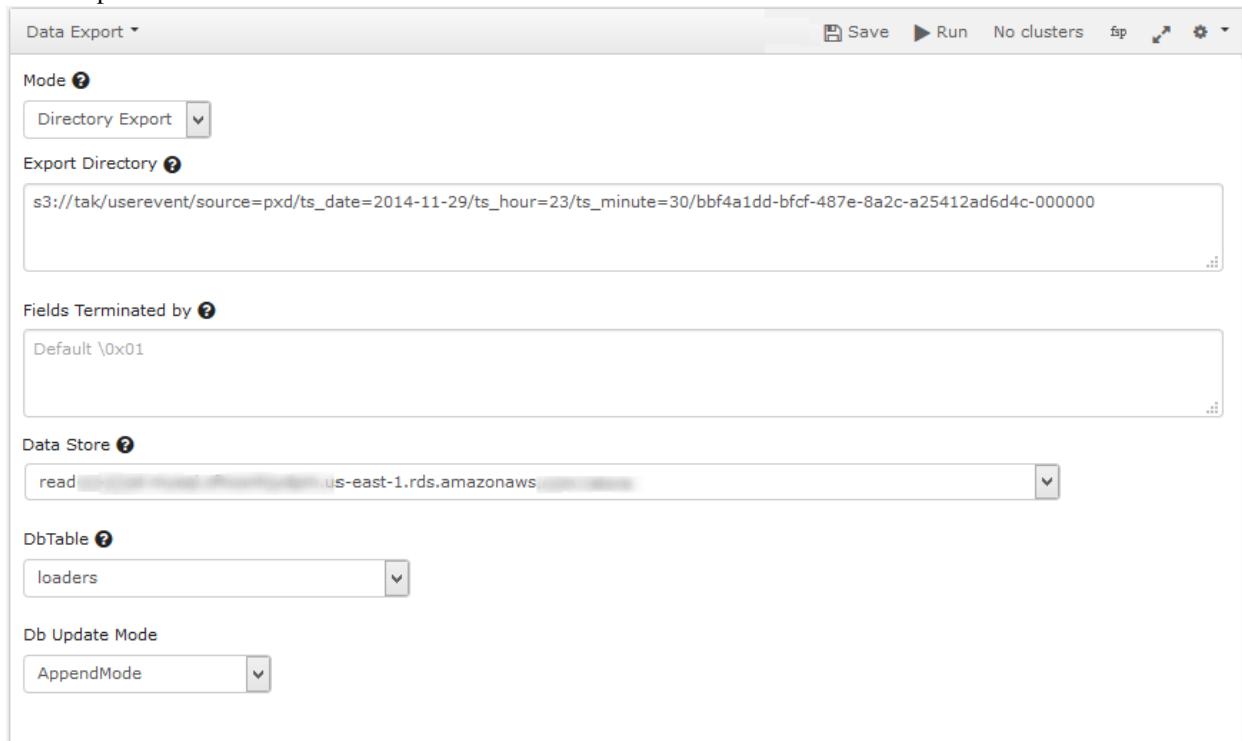
The source directory is also mentioned in the **Export Directory** text field.

4. Type a field separator in the **Field Terminated by** text field if you do not want the default field separator, \0x01.
5. Select a data store from the **Data Store** drop-down list.
6. Once you select the data store, the **DbTable** drop-down list gets loaded with the tables. Select a table from the list.
7. Select a update mode from the **Db Update Mode** drop-down list. **Append** mode is the default selection.

Update and **Insert and Update** are the other two options.

Insert and Update is available only for MySQL database type.

The following figure illustrates the query composer with fields to run a S3 data export query in the directory export mode.



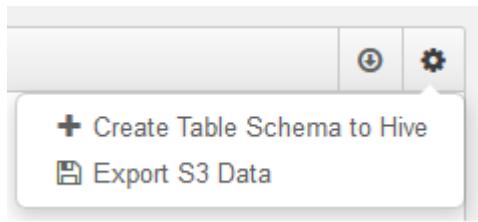
8. Click **Run**. The result is displayed in the **Results** tab.

Creating Schema from Amazon S3 Data

From the sample data in an S3 bucket, you can use **Create Schema** functionality for creating a Hive table in an existing folder/partition. Perform the following steps:

1. Select the file from which you want to create table schema. By default, the **Properties** tab is displayed. Click the **Sample Data** tab. Select the **Format** and **Delimiter** from the drop-down lists. Click the **Create Schema**

button. See *Exploring Data in Amazon S3* for more information. Alternatively, click the icon  to get a drop-down list as shown in the following figure.



Select **Create Table Schema To Hive**.

2. The **Create Table Schema to Hive (1/4)** dialog is displayed. Select the root file directory for the table. Type the partition name in the corresponding text field.

The following figure provides an example of selecting a root directory and naming partitions.

Selected file
s3://ap-dev-qubole/common/hive/30day_1/30daysmall/20100101/EU

Please select the root directory for the table:
common / hive / 30day_1 / 30daysmall / 20100101 / EU / 1.tsv

Partitions Details

Partition 0

Partition 1

Next

Click **Next** to continue.

3. Clicking **Next** displays the **Create Table Schema to Hive (2/4)** dialog. Type a name in the **Hive Table Name** text field. You can change the **Format** and **Delimiter**.

A sample dialog is as shown in the following figure.

The dialog has the following settings:

- Hive Table Name:** New2
- Format:** Text
- Delimiter:** Tab

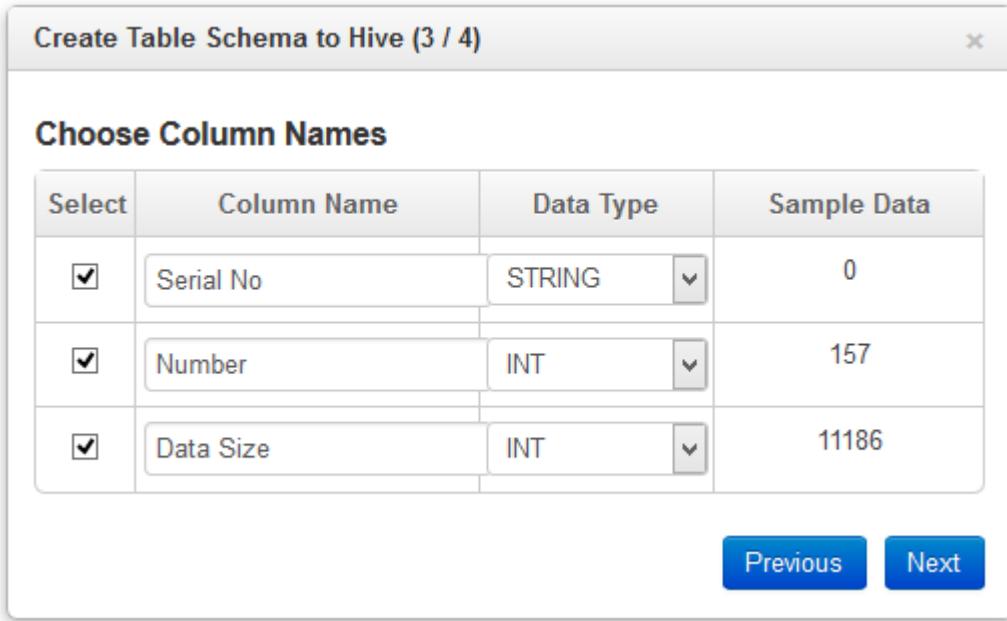
Columns

Col 0	Col 1	Col 2
0	157	11186
1	43	67824
2	15	40880
3	129	50073

Buttons: Previous, Next

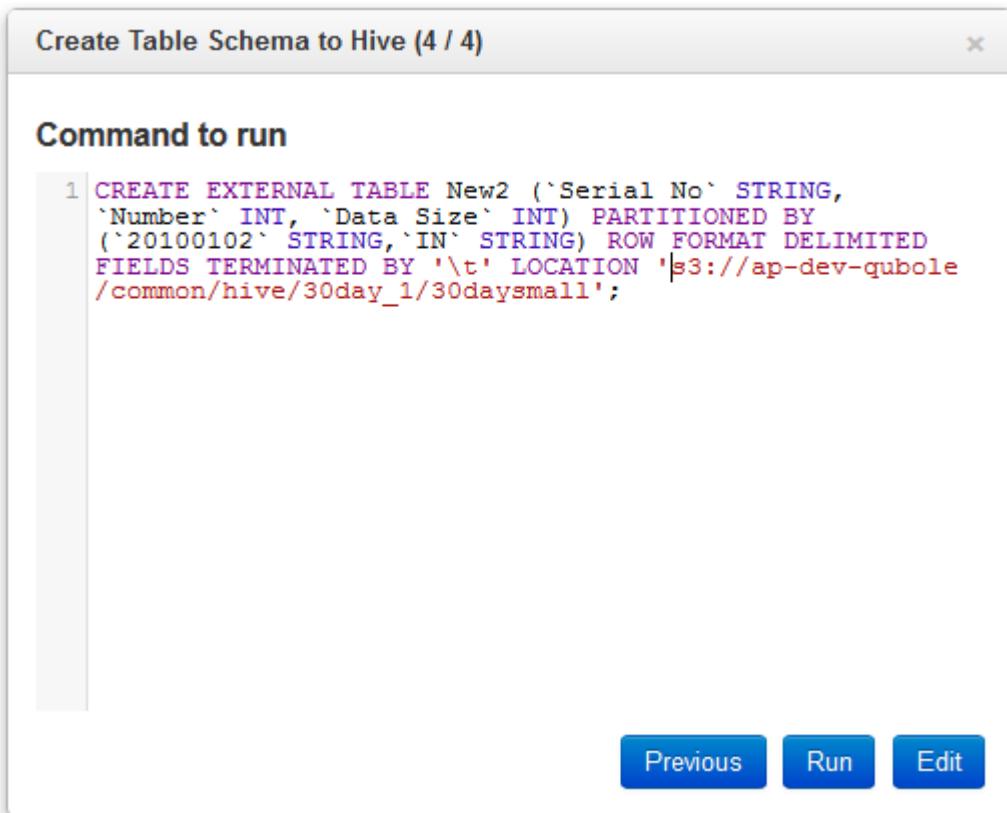
Click **Next** to continue. Click **Previous** to go back to the previous step.

4. Clicking **Next** displays the **Create Table Schema to Hive (3/4)** dialog. It is used to select the columns that will be part of the Hive table selected in the previous step. All column names are selected by default. Columns that are not required can be unchecked. Column names can be changed if required. A sample dialog is as shown in the following figure.



Click **Next** to continue. Click **Previous** to go back to the previous step.

5. Clicking **Next** displays the **Create Table Schema to Hive (4/4)** dialog. This dialog displays the query to create the hive table as illustrated in the following figure.



Click **Previous** to go back to the previous step. Click **Run** to execute the hive query. Click **Edit** to modify the hive query.

Clicking **Run/Edit** opens the query composer of **Analyze** with the Hive query statement. You can modify the query statement before executing it.

In the query composer, click **Run** to execute the hive query.

Exploring Data in Postgres and Redshift Data Stores

Qubole now supports exploring non-default schema in addition to the default schema in Postgres and Redshift data stores as described in:

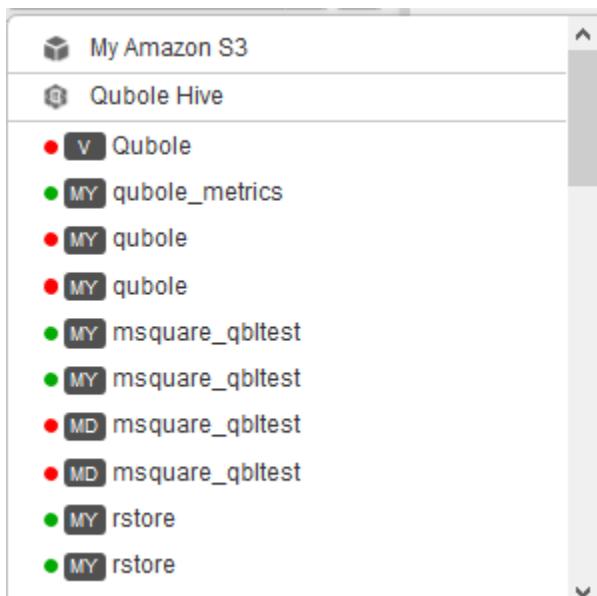
- [Exploring Data in a Postgres Data Store](#)
- [Exploring Data in a Redshift Data Store](#)

Exploring Data in a Postgres Data Store

After adding data into a Postgres data store, navigate to [Explore](#). By default, you see the Qubole Hive database with the expanded default database as shown in the following figure.

The screenshot shows the Qubole Hive interface. At the top left is a dropdown menu labeled "Qubole Hive". Below it is a search bar with "Search in default" and a "Go!" button. On the left, there's a list of schemas: "abcdef", "abcdef1", and "default". The "default" schema is currently selected and expanded, showing two tables: "3int" and "3int_0". To the right of the schema list is a panel titled "Qubole Hive" with tabs for "Properties" and "Access Details". Under "Access Details", it says "Number of Schema(s) 4".

To see the list of data stores, pull down the drop-down list as shown in the following figure.



Select a Postgres data store from the list of data stores as shown in the following figure.

Qubole Hive > pg_catalog

Properties		⚙️
Access Details		
Repository Id:	3590	<input checked="" type="button"/> Edit <input type="button"/> Remove
Repository Name:	d27ofqm6j2gmrn	
Database Type:	postgresql	
Database Name:	d27ofqm6j2gmrn	
Server Address:	ec2-54-204-16-70.compute-1.amazonaws.com	
Port:		
Username:	ethvbjwimpwbuo	
Password:	***** Show password	
Region:	us-east-1	
S3 location for server certificate:		
S3 location for client certificate:		
S3 location for client key:		
Properties		⚙️
Access Details		
Number of Table(s)	77	
Number of View(s)	0	

Edit and **Remove** actions are seen in the list. You can click **Remove** to delete the data store.

Click the **Edit** button and the data store's fields become editable as shown in the following figure.

Properties

Access Details

Database Type	Postgres	
Database Name	d27ofqm6j2gmrn	
Server Address	ec2-54-204-16-70.compute-1.amazonaws.com	
Port	Port	(blank for default)
Username	ethvbjwimpwbuo	
Password	*****	
Region	us-east-1 (North Virginia)	

Qubole supports SSL mode for postgres dbs if its configured on server. Do you want to use it?

Update **Reset to Default** **Cancel**

* Please ensure that our Amazon account id (805246085872) and our security group ('default') have access privileges to your database.
 - If your database is not in VPC then you can give these permissions by adding rules to security group of database.
 - If your database is in VPC use 'on-premise' as database region.
[See here](#) for more information.

Properties

Access Details

Number of Table(s)	77
Number of View(s)	0

Modify the data store fields that you want to and click **Update** to save the changes. Click **Reset to Default** to restore the default settings. Click **Cancel** to retain the previous settings.

Click a table from the non-default database to view its sample data. By default, the **Properties** tab is displayed. Click

 the icon to see the actions.

Import table to Hive is displayed as shown in the following figure.

The screenshot shows a user interface for managing database tables. At the top, there are tabs for 'Rows' and 'Properties'. The 'Properties' tab is selected, showing 'Table Properties' and 'Column Properties' sections. In the 'Table Properties' section, the name is set to 'adv_import_test_table'. In the 'Column Properties' section, there are two columns: 'id' (integer, NO, NULL) and 'data' (character, YES, NULL::bpchar). On the right side of the interface, there are several icons: a checkmark, a refresh, a gear, and a 'Import table to Hive' button.

Click the **full view icon** to see the table in a full window.



Click the **refresh icon** to refresh the table.



Clicking **Import table to Hive** displays the query composer of **Analyze** in another tab. The query composer contains a data import query with the Postgres data store selected in the **Data Store** drop-down list. See [Composing a Data Import Query](#) for more information on a data import query. Hive table and related text fields contain the default values as shown in the following figure.

Data Import ▾

Save Run No clusters Ftp ▾

Data Store ?
ec :-1.amazonaws

DB Schema ?
public

DB Table ?
adv_import_test_table

Columns to Extract
Select columns to import

Filters
Optional where clause eg: id > 10

Parallelism ?
Automatic

Mode
Simple

Hive Database ?
default

Hive Table Name ?
adv_import_test_table

Choose output table format
text

Hive Table Partition Spec ?

Click **Rows** to see the sample data in the table as illustrated in the following figure.

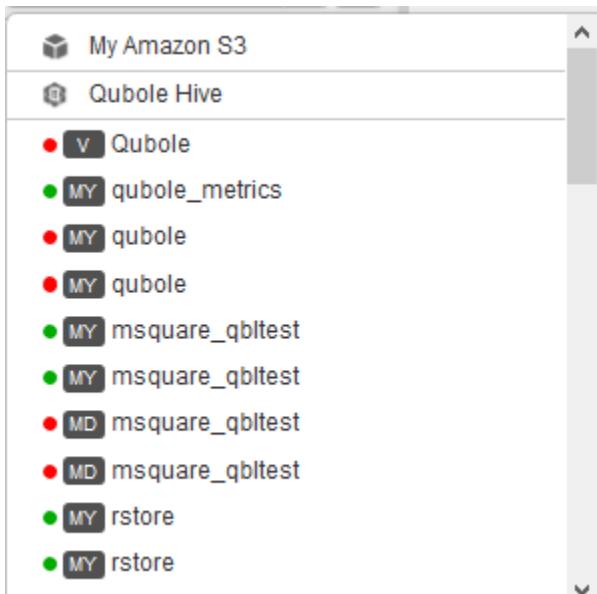
Rows	Properties
Rows Properties Export Copy Settings	
id	data
1	one
2	two
3	three
4	four
5	five

Exploring Data in a Redshift Data Store

After adding data into a Redshift data store, navigate to [Explore](#). By default, you see the Qubole Hive database with the expanded default database as shown in the following figure.

The screenshot shows the Qubole Hive interface. At the top left is a dropdown menu set to "Qubole Hive". To its right are a search bar with "Search in default" and a "Go!" button. Below the search bar is a list of data stores: "abcdef", "abcdef1", "default" (which is selected and highlighted in blue), and "3int", "3int_0". On the right side, under the heading "Qubole Hive", is a "Properties" tab. Below it, the "Access Details" section displays "Number of Schema(s)" as 4.

To see the list of data stores, pull down the drop-down list as shown in the following figure.



Select a Redshift data store from the list of data stores and it is as shown in the following figure.

Qubole Hive > pg_catalog

Properties	
Access Details	
Repository Id:	3591
Repository Name:	qubole
Database Type:	redshift
Database Name:	qubole
Server Address:	quboletest.cqye4rtfs8hv.us-east-1.redshift.amazonaws.com
Port:	
Username:	qubole
Password:	***** Show password
Region:	us-east-1

Properties	
Access Details	
Number of Table(s)	162
Number of View(s)	0

Edit and **Remove** actions are seen in the list. You can click **Remove** to delete the data store.

Click the **Edit** button and the data store's fields become editable as shown in the following figure.

Properties

Access Details

Database Type: Redshift

Database Name: qubole

Server Address: quboletest.cqye4rtfs8hv.us-east-1.redshift.amazonaws.com

Port: Port (blank for default)

Username: qubole

Password: *****

Region: us-east-1 (North Virginia)

Update **Reset to Default** **Cancel**

* Please ensure that our Amazon account id (805246085872) and our security group ('default') have access privileges to your database.
- If your database is not in VPC then you can give these permissions by adding rules to security group of database.
- If your database is in VPC use 'on-premise' as database region.
[See here](#) for more information.

Properties

Access Details

Number of Table(s): 162

Number of View(s): 0

Modify the data store fields that you want to and click **Update** to save the changes. Click **Reset to Default** to restore the default settings. Click **Cancel** to retain the previous settings.

Click a table to from the database to view its sample data. By default, the **Properties** tab is displayed. Click the icon



to see the actions.

Import table to Hive is displayed as shown in the following figure.

Table Properties

Name: pg_aggregate

Column Properties

Column	Data Type	Null	Key	Default
aggfinalfn	regproc	NO		
aggtransfn	regproc	NO		
aggnoid	regproc	NO		
agginitalval	text	YES		
aggtranstype	oid	NO		

Click the **full view icon**  to see the table in a full window.

Click the **refresh icon**  to refresh the table.

Clicking **Import table to Hive** displays the query composer of **Analyze** in another tab. The query composer contains a data import query with the Redshift data store selected in the **Data Store** drop-down list. See [Composing a Data Import Query](#) for more information on a data import query. Hive table and related text fields contain the default values as shown in the following figure.

Data Import ▾

Save Run No clusters fp

Data Store ?
qubole redshift.amazonaws

DB Schema ?
pg_catalog

DB Table ?
pg_aggregate

Columns to Extract
Select columns to import

Filters
Optional where clause eg: id > 10

Parallelism ?
Automatic

Mode
Simple

Hive Database ?
default

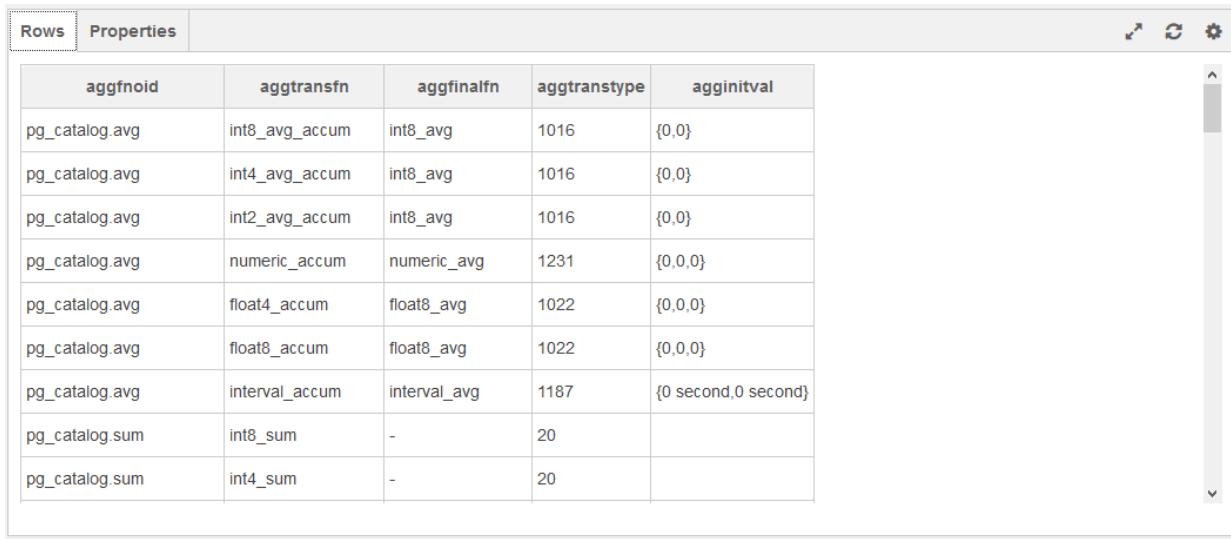
Hive Table Name ?
pg_aggregate

Choose output table format
text

Hive Table Partition Spec ?



Click **Rows** to see the sample data in the table as illustrated in the following figure.



aggnoid	aggtransfn	aggfinalfn	aggtranstype	agginitalval
pg_catalog.avg	int8_avg_accum	int8_avg	1016	{0,0}
pg_catalog.avg	int4_avg_accum	int8_avg	1016	{0,0}
pg_catalog.avg	int2_avg_accum	int8_avg	1016	{0,0}
pg_catalog.avg	numeric_accum	numeric_avg	1231	{0,0,0}
pg_catalog.avg	float4_accum	float8_avg	1022	{0,0,0}
pg_catalog.avg	float8_accum	float8_avg	1022	{0,0,0}
pg_catalog.avg	interval_accum	interval_avg	1187	{0 second,0 second}
pg_catalog.sum	int8_sum	-	20	
pg_catalog.sum	int4_sum	-	20	

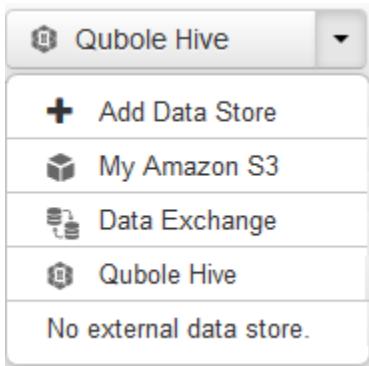
Adding Data Exchange Spaces

In Data Exchange, *Space* is a logical storage to which you can export the table data that is stored in an Amazon S3 location. The table's metadata such as column information and data location at the time of export is captured and saved in the Amazon S3 location. The third-party data analysts and partners can access the table data you have shared by subscribing to a *Space* and import the data to do analysis. See [Data Exchange](#) for more information.

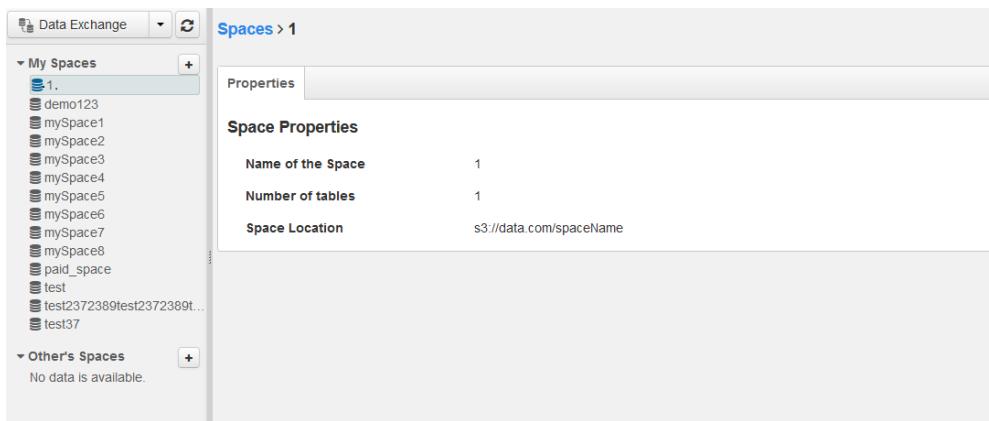
Creating a Space

A Space is a logical storage created to share the Hive table location to others, who want to access that data. Spaces are listed in **My Spaces** in the user interface. To create a Space, perform the following steps:

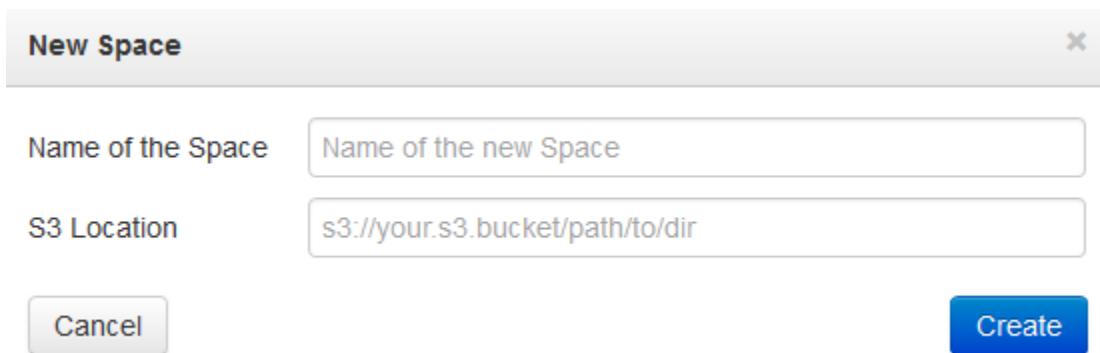
1. Navigate to [Explore](#).
2. Pull the drop-down list that contains **Qubole Hive** selected by default as shown in the following figure.



3. Click **Data Exchange**. If there is any existing Space, it is displayed in **My Spaces** as shown in the following figure.



4. Click the + sign to add a **Space**. The **New Space** dialog is displayed as shown in the following figure.



Enter a name in the **Name of the Space** text field. Provide the S3 location of Hive table data that you want to share in the **S3 Location** text field.

Click **Create** to add the **Space**. Click **Cancel** if you do not want to add it.

Subscribing to a Space

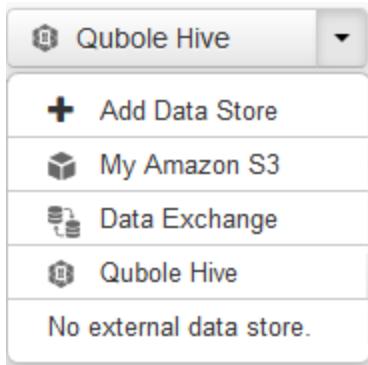
A partner can subscribe to a Space by providing access details for that Space. Subscribed Spaces are listed in **Other's Space** section of the user interface. A partner can then navigate to the subscribed Space to see exported tables and import those tables into his account and access the Hive table data to do further analysis.

Prerequisites:

You must provide the name of Space, AWS Role ARN, and AWS External ID to the partner to subscribe to a particular Space using the AWS Console.

Perform the following steps to subscribe to a Space:

1. Navigate to [Explore](#).
2. Pull the drop-down list that contains **Qubole Hive** selected by default as shown in the following figure.



3. Click **Data Exchange**. If there is any existing subscribed space, it is displayed in **Other's Spaces** as shown in the following figure.

The screenshot shows the 'Data Exchange' interface with a sidebar. Under 'MY Spaces', there is a list of spaces including '1.', 'demo123', 'mySpace1', 'mySpace2', 'mySpace3', 'mySpace4', 'mySpace5', 'mySpace6', 'mySpace7', 'mySpace8', 'paid_space', 'test', 'test2372389test2372389t...', and 'test37'. Under 'Other's Spaces', it says 'No data is available.' On the right, a panel titled 'Spaces > 1' shows 'Space Properties' for a single space: Name of the Space (1), Number of tables (1), and Space Location (s3://data.com/spaceName).

4. Click the + sign to subscribe to a Space. The **Add Other's Space** dialog is displayed as shown in the following figure.

The dialog box has a title bar 'Add Other's Space' with a close button. It contains three input fields: 'Name of the Space' (text box), 'AWS Role ARN' (text box with placeholder 'E.g. arn:aws:iam::xxyyzzz:role/AaaBbbCcc'), and 'AWS External ID' (text box). At the bottom are two buttons: 'Cancel' (gray) and 'Add Space' (blue).

As a partner, add the name of the space that you have access to in the **Name of the Space** text field. Enter the corresponding values of the partner/third-party user in **AWS Role ARN** and **AWS External ID** text fields.

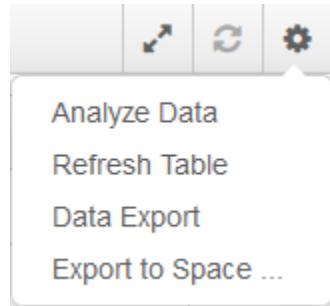
Click **Add Space** to add it to the list of subscribed spaces listed within **Other's Spaces** in the user interface. Click **Cancel** if you do not want to add it.

Managing My Spaces

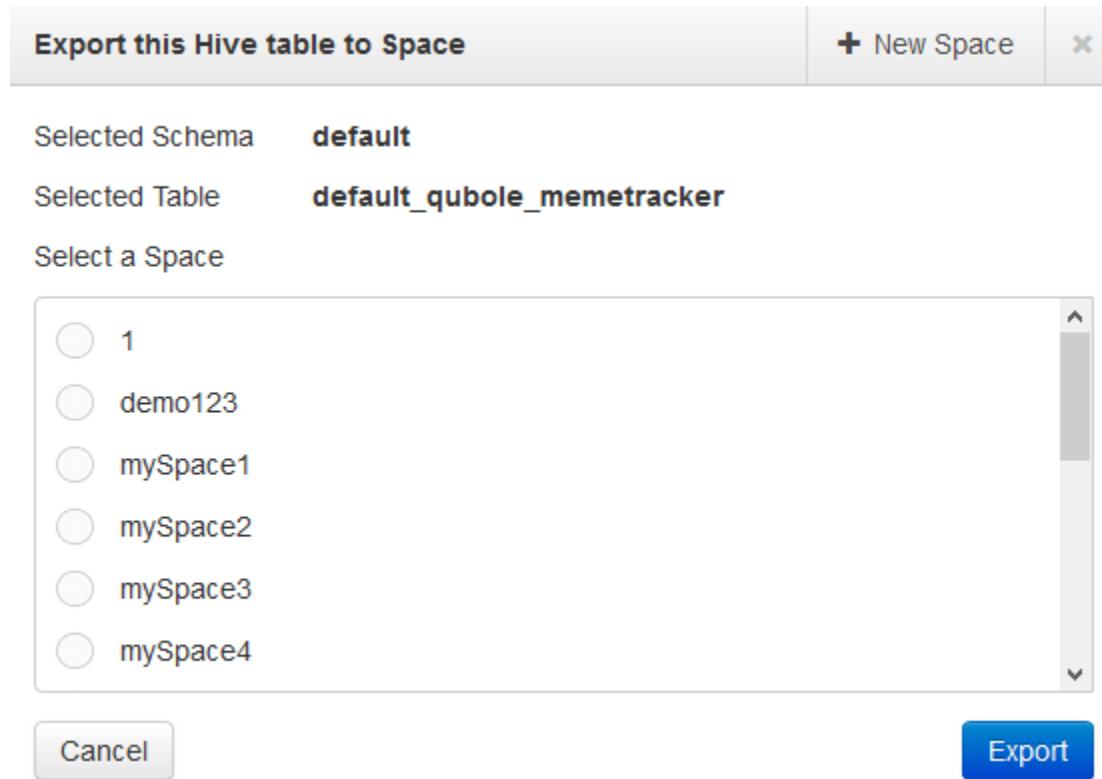
Creating a Space describes how to add a Space to share hive table data. Export Hive table data from the S3 location to a Space through the Qubole Hive metastore. See *Data Exchange* for more information.

Navigate to [Explore](#) and in the Qubole Hive metastore, select the Hive table and click the icon .

A drop-down list is displayed as shown in the following figure.



Click **Export to Space** from the list. The **Export this Hive table to Space** dialog is displayed as shown in the following figure.



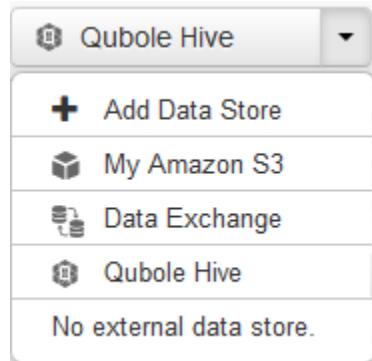
Select a **Space** and click **Export**. Click **Cancel** if you do not want to export the data.

Click **+ New Space** if you want to create a new space and export the Hive table data into it.

Creating a Space describes how to create a new Space.

Managing Data in Spaces

In the **Explore** tab, pull the drop-down list that contains **Qubole Hive** selected by default as shown in the following figure.



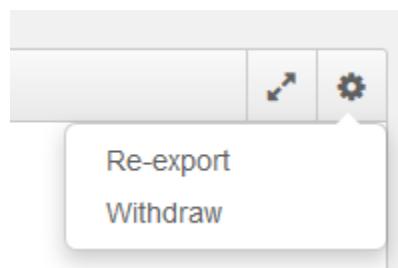
Select **Data Exchange** to see the list of spaces and other's spaces.

To manage the Hive table data in a specific Space, click that space and you can see the table properties as shown in the following figure.

Column	Data Type	Comments
site	string	from deserializer
ts	string	from deserializer
phr	string	from deserializer
lnks	string	from deserializer
month	string	

Click the settings icon

The list of actions is displayed as shown in the following figure.



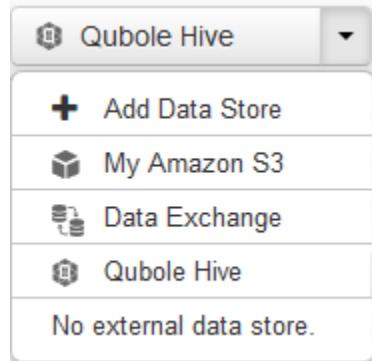
If a Hive table's data is modified after it is exported to a Space, click **Re-export** to notify the partner to re-import the updated Hive table.

If you do not want the Hive table to be shared, click **Withdraw**.

Managing Other's Spaces

Other's Spaces are created and Hive table data in the Space is imported to a particular hive schema selected by a partner for further data analysis. *Subscribing to a Space* describes how to add an other's space. See *Data Exchange* for more information.

Navigate to **Explore** and pull the drop-down list that contains **Qubole Hive** selected by default as shown in the following figure.



Select **Data Exchange** to see the list of spaces and other's spaces.

Click a space from the other's spaces' list (if any) that you want to manage. The following figure illustrates a subscribed space.

The screenshot shows a user interface for managing a subscribed space named "2322Test".

Space Properties:

Name of the Space	2322Test
Number of tables	3

Tables:

Table Name	Imported Schema	Data Location	Actions
default_qubole_airline_origin_destination	default	s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination	[down arrow]
default_qubole_memetracker	-	s3n://paid-qubole/default-datasets/memetracker	[down arrow]

The green-tick mark icon against a Hive table indicates that it is successfully imported.

Clicking the downward arrow in the **Actions** column displays **Remove** action as shown in the following figure.

[Other's Spaces > 2322Test](#)

Properties															
Space Properties															
Name of the Space	2322Test														
Number of tables	3														
<table border="1"> <thead> <tr> <th>Table Name</th> <th>Imported Schema</th> <th>Data Location</th> <th>Actions</th> </tr> </thead> <tbody> <tr> <td>default_qubole_airline_origin_destination</td> <td>default</td> <td>s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination</td> <td></td> </tr> <tr> <td>default_qubole_memetracker</td> <td>-</td> <td>s3n://paid-qubole/default-datasets/memetracker</td> <td></td> </tr> </tbody> </table>				Table Name	Imported Schema	Data Location	Actions	default_qubole_airline_origin_destination	default	s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination		default_qubole_memetracker	-	s3n://paid-qubole/default-datasets/memetracker	
Table Name	Imported Schema	Data Location	Actions												
default_qubole_airline_origin_destination	default	s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination													
default_qubole_memetracker	-	s3n://paid-qubole/default-datasets/memetracker													

Click **Remove** if you want to remove the Hive table.

Click the downward arrow in the **Actions** column for the Hive table that does not have a green icon against it. The **Import** action is displayed as shown in the following figure.

[Other's Spaces > 2322Test](#)

Properties															
Space Properties															
Name of the Space	2322Test														
Number of tables	3														
<table border="1"> <thead> <tr> <th>Table Name</th> <th>Imported Schema</th> <th>Data Location</th> <th>Actions</th> </tr> </thead> <tbody> <tr> <td>default_qubole_airline_origin_destination</td> <td>default</td> <td>s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination</td> <td></td> </tr> <tr> <td>default_qubole_memetracker</td> <td>-</td> <td>s3n://paid-qubole/default-datasets/memetracker</td> <td></td> </tr> </tbody> </table>				Table Name	Imported Schema	Data Location	Actions	default_qubole_airline_origin_destination	default	s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination		default_qubole_memetracker	-	s3n://paid-qubole/default-datasets/memetracker	
Table Name	Imported Schema	Data Location	Actions												
default_qubole_airline_origin_destination	default	s3n://paid-qubole/default-datasets/transportation/aviation/airline_origin_destination													
default_qubole_memetracker	-	s3n://paid-qubole/default-datasets/memetracker													

Click **Import** to import the Hive table from the shared Space.

If a Hive table that is already imported into an other's space, has changed at its S3 location and has been re-exported to the Space, then an update icon is marked against such Hive table.

Click downward arrow in the **Actions** column for such Hive table to see two actions as shown in the following figure.

Table Name	Imported Schema	Data Location	Actions
demo_test_table	default	s3n://data.com/job_test_data	
job_table	-	s3n://data.com/job_test_data	
job_test_table	default	s3n://data.com/job_test_data	
job_test_table	default	s3n://data.com/job_test_data	
new_test_table	default	s3n://data.com/job_test_data	
space_test_table	database1	s3n://data.com/job_test_data	

Click **Re-import** to get the updated Hive table. Click **Remove** to delete the Hive table from the space.

The above figure also shows the table update icon and the table import icon along with the red cross-mark icon , which indicates that the Hive table is removed (dropped) from the native Hive schema.

If you click the downward arrow in the **Actions** column for a Hive table that has the red cross-mark icon against it, you can see the **Remove** action. You can click it to remove the Hive table from the **Other's Space**.

Editing an Other's Space

To change a partner's **AWS Role ARN** and **AWS External ID** credentials, click that Other's Space. In the **Other's Space** dialog, click the edit button .

The **Edit Space** dialog is displayed as shown in the following figure.

Name of the Space	mySpace7
AWS Role ARN	E.g. arn:aws:iam::xxxyyzzz:role/AaaBbbCcc
AWS External ID	AWS External ID

As a partner, modify the **AWS Role ARN** and **AWS External ID** credentials provided to you and click **Update**. Click **Cancel** to retain the existing values.

2.2.3 Analyze

About the Analyze User Interface

Analyze is the browser-based UI front end to interact with the Qubole Data Service (QDS). Analyze can be used to author and execute Hive, Hadoop, Pig, Shell, and other commands on datasets. You can also use Analyze to exchange data between various databases and Qubole using the Data Import Commands and Data Export Commands. This section walks through the various features and tools available via the tabs of the Analyze page. The Analyze page contains three tabs and a button:

- *Compose (button)*
- *History*
- *Repo*
- *Tables*
- *S3*

History Tab

Navigating to the Analyze page displays the History tab by default as shown in the following figure.

The screenshot shows the Analyze History tab interface. At the top, there are five tabs: Compose, History (which is selected), Repo, Tables, and S3. Below the tabs is a search bar with the placeholder "interface:API,UI,TEMPLATE,ODBC,SMART_QUERY qbol_user_id:qubole@qubole.com". The main area displays a list of command history entries. Each entry includes a timestamp, a user icon, a command ID, a star icon, the command text, and a "Hive Query" dropdown menu. The entries are as follows:

- 5166151 ☆
curl -X POST -H "X-AUTH-TOKEN: \$AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" \ -d '{ "query": "show tables;" "label": "1" }' \ "https://api.qub...
- 5166139 ☆
curl -i -X POST -H "X-AUTH-TOKEN: \$AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" \ -d '{ "query": "show tables;" "label": "1" }' \ "https://api....
- 5166132 ☆
curl -i -X POST -H "X-AUTH-TOKEN: \$AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" \ -d '{ "query": "show tables;" "label": "1" }' \ "https://api....
- 5048801 ☆
CREATE EXTERNAL TABLE New2 ('Col0' STRING, `Col1` STRING, `Col2` STRING) PARTITIONED BY ('20100102' STRING,`IN` STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' LOCATION 's3://ap-dev-qubole/comm...
- 5010859 ☆
CREATE EXTERNAL TABLE S3HiveSample ('Export1' STRING, `Export2` STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

This tab can be used to browse the previous commands, logs and results. This tab provides basic query authoring and execution interface, which can be used to edit the previous queries and rerun them. This tab also allows to search for previously run commands by providing the command ID, command type, status, user, tags, name , and so on. It also allows scheduling these commands.

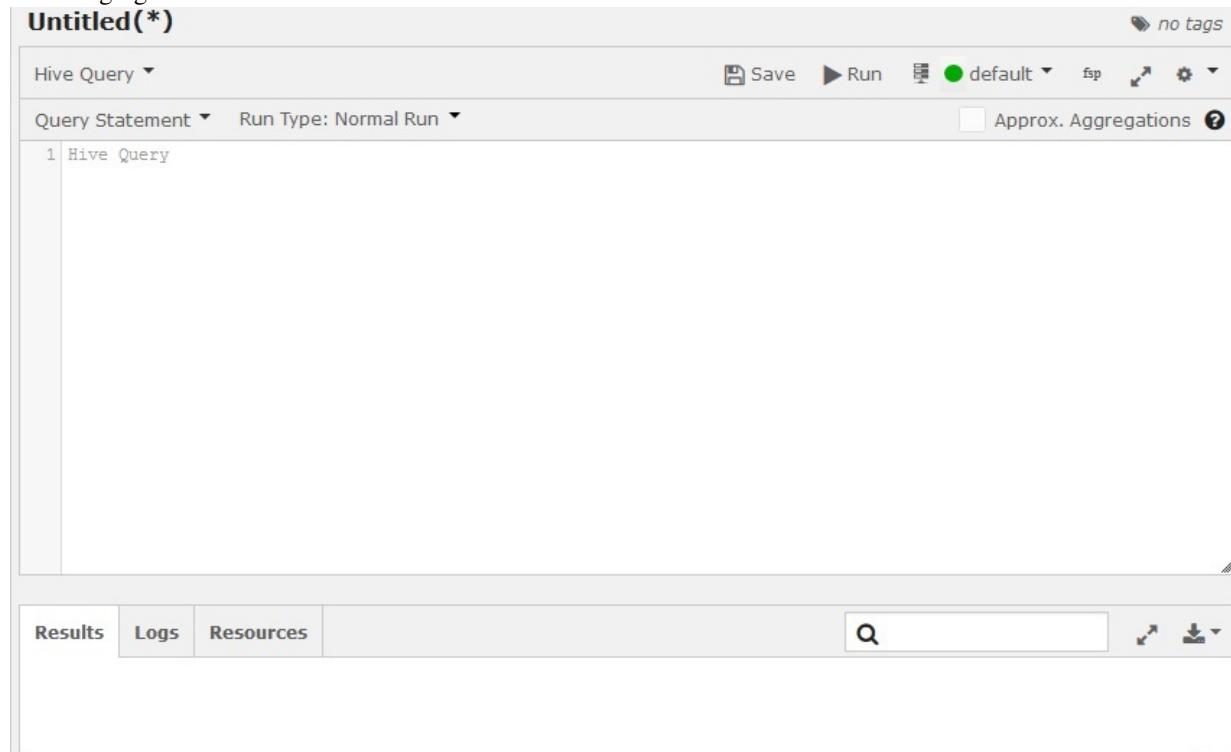
Compose

Clicking the **Compose** button displays a query composer and editor, which is used to author and execute these type of commands:

- Data Export
- Data Import
- DB query

- Hive Query
- Hadoop Job
- Pig Query
- Shell Command
- Presto Query
- Spark Command
- Redshift Query
- Query Export
- Refresh Table
- Workflow

By default, the **Compose** query composer is displayed with Hive query selected as the command type, as shown in the following figure.



This is a command editor window, which allows composing various kinds of commands. It allows running commands and also see **Results** and **Logs** for the runs as it progresses. The command editor has usability and developer productivity features such as highlighted syntax and auto-completion. The commands run can be given a name as well as tagged with multiple tags. The command editor also allows choosing the cluster against which the command should be run. Hive and Presto Queries can also use an expression editor which allows evaluating an expression for a table or a column and shows the result. This can significantly cut down the time required to author queries because it reduces the time needed to discover bad expressions in a query. A typical workflow would involve you finessing the expressions that appear in queries through fast iterations in the Expression Evaluator before submitting a query to a Qubole cluster. Hive queries can be run in normal, constrained, and test mode as well.

The command editor/query composer also contains a **Resources** tab that is next to the **Logs** tab. The **Resources** tab contains the Job Tracker URLs for Hadoop-1 jobs, Application Tracker URL for Hadoop-2 and Spark jobs, and Spark Application UI URLs for Spark queries. In general, the Job Tracker URLs are displayed for Hive and Spark queries.

An example of Job Tracker URLs displayed in the **Resources** tab is shown in the following figure.

Job Tracker

1. Application UI
[https://\[REDACTED\].et/cluster-proxy?encodedUrl=http...](https://[REDACTED].et/cluster-proxy?encodedUrl=http...)
2. Spark Application UI
[https://\[REDACTED\].et/cluster-proxy?encodedUrl=http...](https://[REDACTED].et/cluster-proxy?encodedUrl=http...)

See Quick Start Guide for Hive and Hadoop command examples.

Repo Tab

This tab contains saved queries from the compose tab. You can save a query after creating/running it in the query composer. The saved queries are maintained in the **Repo** tab.

An example hive query is as shown in the following figure.

Hive Query

Query Statement: `1 select * from default_qubole_airline_origin_destination`

Run Type: Normal Run

Save Run Default FSP Approx. Aggregations

In the query composer, after running a query, click **Save** if you want to rerun it again or refer to it in future. After the **Save** button is clicked, the **Query Details** dialog is displayed. Enter a name for the query. An example of the Query Details dialog is as shown in the following figure.

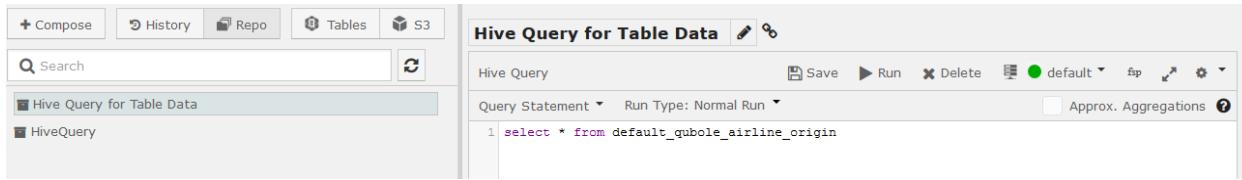


The saved queries are maintained in the **Repo** tab as shown in the following figure.

The screenshot shows the Qubole Data Service interface with the "Repo" tab selected. The top navigation bar includes "Compose", "History", "Repo" (selected), "Tables", and "S3". Below the navigation is a search bar with the placeholder "Search" and a refresh icon. The main area displays a list of saved queries. The first query, "Hive Query for Table Data", is highlighted with a blue background. Below it, another entry "HiveQuery" is visible.

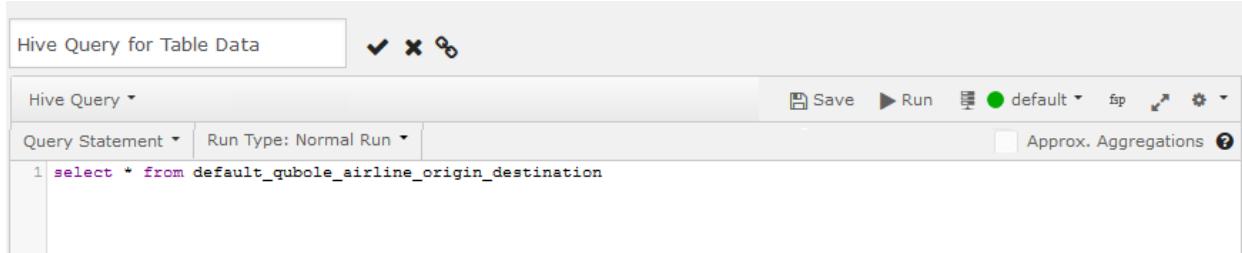
To change the query name, click the query from the **Repo** tab and the same query gets displayed in the query composer.

Do a mouse hover on the query name and an edit icon is visible as shown in the following figure.



Click the **permalink** icon  to see the query's permalink.

Click the edit icon and the query name field becomes editable as shown in the following figure.



After changing the query name, click the check mark symbol  to save it.

Click the cross mark symbol  to cancel saving the new query name.

For each saved query, you can see the date and time at which that specific query was run. Click a query in the **Repo** tab and see the **Commands** tab that is below the query composer as shown in the following figure.

The screenshot shows the Qubole interface with the 'Commands' tab selected. Above the table, the query 'select * from default_qubole_airline_origin_destination' is displayed. The 'Commands' table has columns: id, command type, user, and time. One row is shown:

id	command type	user	time
7350010	HiveCommand	Qubole J	19/06/15 10:00

You can edit the saved query and save it again. Saving the edited query creates different versions that are maintained in the **Versions** tab as shown in the following figure.

The screenshot shows the Qubole Data Service interface for managing Hive queries. At the top, there's a header bar with 'Hive Query for Table Data' and various navigation icons. Below it is a 'Hive Query' editor window containing a single line of code: 'select * from default_qubole_airline_origin'. The editor also includes a 'Save' button, a 'Run' button, and a dropdown for selecting a database ('default'). There's a checkbox for 'Approx. Aggregations' and a help icon. Below the editor is a table titled 'Commands' showing two previous queries:

query	command type	created time
select * from default_qubole_airline_origin	HiveCommand	19/06/15 12:07
select * from default_qubole_airline_origin_destination	HiveCommand	17/06/15 11:39

Tables Tab

This tab allows viewing metadata related to Hive Tables in your account. This tab lists all the schemas and tables in Hive and allows viewing columns and their types. This tab also allows refreshing the schema to get metadata about any newly created tables in Hive. This view helps in composing hive or presto queries against the hive tables.

The default view of **Tables** is as shown in the following figure.

The screenshot shows the 'Tables' tab in the Qubole interface. It features a search bar at the top and a sidebar with buttons for 'Compose', 'History', 'Repo', 'Tables', and 'S3'. The main area displays a tree view of tables under the 'default' schema. The 'default' schema contains several tables, each with a list of columns. Some tables have a plus sign next to them, indicating they can be expanded to view more details.

- adas
- b1b2
- datalogix
- default**
 - 3int_100m_sqooped
 - 3int_200k_limited
 - 3int_export
 - 3int_export2
 - 3int_sqooped_2
 - 3int_t1
 - a123
 - abc
 - abc123
 - abcaabc
 - abcabc123
 - accounts

S3 Tab

This tab allows browsing Amazon S3 buckets and viewing the directories and files.

Mapping of Cluster and Command Type

QDS supports a few command types on certain clusters. The following table shows the mapping of command types and clusters.

Command Type	Hadoop-1 Cluster Support	Hadoop-2 Cluster Support	HBase Cluster Support	Presto Cluster Support	Spark Cluster Support
Data Export	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
Data Import	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
DB Query	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
Hadoop command	Yes	Yes	Yes	Yes	No
Hive query	Yes	Yes	No	Yes	No
Pig query	Yes	Yes	No	Yes	No
Presto query	No	No	No	Yes	No
Query Export	Yes	Yes	No	Yes	No
Redshift command	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
Refresh Table command	Not Applicable	Not Applicable	Not Applicable	Not Applicable	Not Applicable
Shell command	Yes	Yes	Yes	Yes	Yes
Spark command	No	No	No	No	Yes

Note: Cluster support for the workflow command depends on the type of commands in the workflow. DB Query, Data Export, Data Import, Redshift, and Refresh Table commands run without a cluster.

Composing a Data Export Query

Compose a data export query using the query composer and editor available in the **Analyze** page. Data export query can be composed for exporting a hive table and directory from an S3 bucket. See [Data Export](#) for more information on exporting data from hive table and S3 bucket directory.

Prerequisites

You must have an existing data store to which data is to be exported. Create a data store in **Explore** if it does not exist.

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support Db queries and the queries can run without clusters too. See [Mapping of Cluster and Command Type](#) for more information.

Composing a Hive Table Export Query

Perform the following steps to compose a Hive table data export query :

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Data Export** from the **Command Type** drop-down list. An illustration of the compose editor with **Data Export** as the command type is as shown in the following figure.

2. To export hive table, the **Mode** is HiveTableExport (selected by default).
3. Specify the Hive table in the **HiveTable** text field.
4. Specify the hive partition if the hive table has a partition spec in the **Hive Table Partition Spec** text field. This field can be left blank if there is no partition in the hive table.
5. Select a data store from the **Data Store** drop-down list.
6. Once you select a data store, tables get populated in the **DbType** drop-down list. Select a table from the list.
7. In the **DB Update Mode** field, by default, **Append Mode** is selected. The other two options are: **Update Only Mode** and **Insert and Update Mode** (supported for only Oracle MySQL database).
8. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
9. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

[Composing a Directory Export Query](#)

Perform the following steps to export a directory from a S3 bucket:

The steps 1, 5 to 9 as explained in the [Composing a Hive Table Export Query](#) section hold good for composing a directory export query.

There are changes in the steps 2, 3, and 4 as described below:

2. To export a directory from an S3 bucket, select the mode as **Directory Export**.
3. In the **Export Directory** text field, specify the path of the S3 directory.

-
4. In the **Fields Terminated by** text field, you can specify a different value. The query takes the default *0x01* if nothing is specified.

Composing a Data Import Query

You can import data from supported databases, Redshift, Vertica, Microsoft SQL Server, Oracle MySQL, Postgres, and Mongo DB into a Hive table in the Hive metastore using the query composer available in **Analyze**. See [Data Import](#) for more information on importing data into Hive tables.

Prerequisites

You must have an existing data store of the same database type from which data is to be imported. Create a data store in **Explore** if it does not exist.

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support Db queries and the queries can run without clusters too. See [Mapping of Cluster and Command Type](#) for more information. Qubole supports importing non-default schema from Postgres and Redshift data stores into a Hive database.

Compose a Data Import Query

Perform the following steps to compose a data import query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Data Import** from the **Command Type** drop-down list. An illustration of the query composer with **Data Import** as the command type is as shown in the following figure.

The screenshot shows the Qubole Data Import interface. At the top, there are buttons for Save, Run, and No clusters, along with a file icon and a gear icon. The main area is divided into several sections:

- Data Store**: A dropdown menu currently set to "aws-east-1.rds.amazonaws".
- DB Table**: A dropdown menu currently set to "Select".
- Columns to Extract**: A text input field containing "Select columns to import".
- Filters**: A text input field containing "Optional where clause eg: id > 10".
- Parallelism**: A dropdown menu currently set to "1".
- Mode**: A dropdown menu currently set to "Simple".
- Hive Database**: A dropdown menu currently set to "default".
- Hive Table Name**: A dropdown menu currently set to "Select Table".
- Choose output table format**: A dropdown menu currently set to "orc".
- Hive Table Partition Spec**: An empty text input field.

2. Select a data store from the **Data Store** drop-down list.
3. Once you select a data store, tables get populated in the **DbTable** drop-down list. Select a table from the list.
4. In the **Columns to Extract** text field, columns get populated, select columns that you want to extract.
5. In **Filters** column, specify any condition if you want it to be applied to the table data. For example, `id > 5`.
6. Set the parallelism value in the drop-down list. 1 is the default value. See [Data Import](#) for more information on parallelism.
7. Select a mode for importing data. Simple is selected in the **Mode** drop-down list that contains Advanced as the other option. See [Data Import](#) for more information on the data import modes.
8. Select a Hive database from the **Hive Database** drop-down list.
9. Select a Hive table to which data is imported from the **Hive Table Name** drop-down list.
10. Select an output table format from the **Choose output table format** drop-down list. Qubole supports, Avro,

Optimized Row Columnar (ORC), and text formats. Orc is the default table format. See [Avro Tables](#) and [ORC Tables](#) for more information on the the table formats.

11. Specify the hive partition if the hive table has a partition spec in the **Hive Table Partition Spec** text field. This field can be left blank if there is no partition in the hive table.
12. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
13. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

Composing a Db Query

You can query an existing data store by using the query composer available in the **Analyze** page.

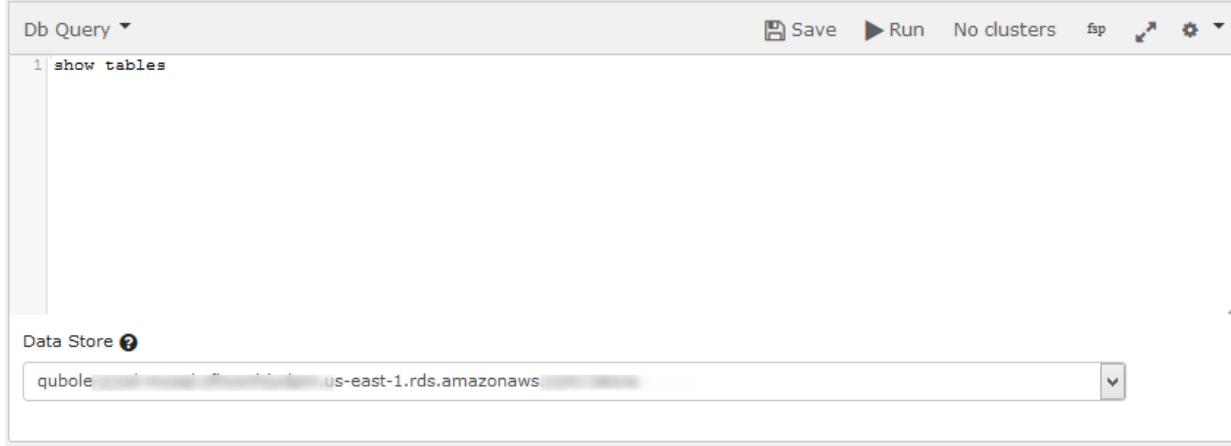
Prerequisites

You must have an existing data store to query it. Create a data store in **Explore** if it does not exist.

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support Db queries and the queries can run without clusters too. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to query an existing data store:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Db Query** from the **Command Type** drop-down list. An illustration of the query composer with **Db Query** as the command type is as shown in the following figure.



2. Enter a query in the text field.
3. Select a data store to which the query is to be applied.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

Composing a Hadoop Job Query

You can compose a Hadoop job query using the query composer available in the **Analyze** page. See [Running a Hadoop Job](#) for more information.

You can use the query composer to three Hadoop job types:

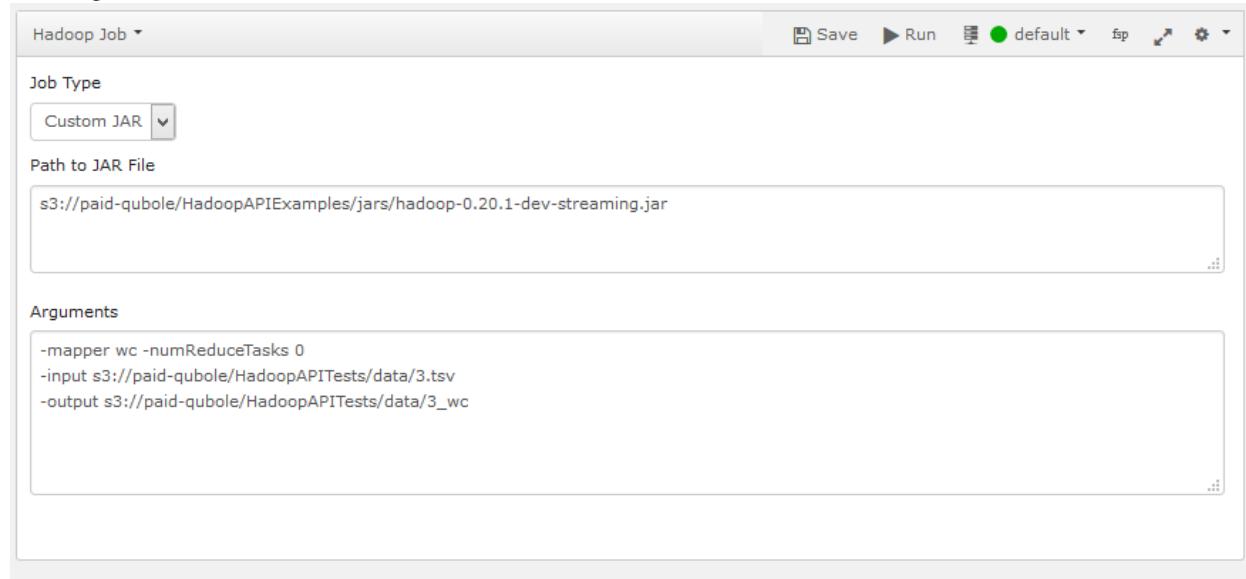
- Custom jar. See [Compose a Hadoop Custom Jar Query](#).
- Streaming. See [Compose a Hadoop Streaming Job](#).
- S3DistCp. See [Compose a Hadoop S3DistCp Command](#).

Note: Hadoop-1, Hadoop-2, HBase, and Presto clusters support Hadoop job queries. See [Mapping of Cluster and Command Type](#) for more information.

Compose a Hadoop Custom Jar Query

Perform the following steps to compose a Hadoop jar query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Hadoop Job** from the **Command Type** drop-down list. An illustration of the compose editor with **Hadoop Job** as the command type is as shown in the following figure.



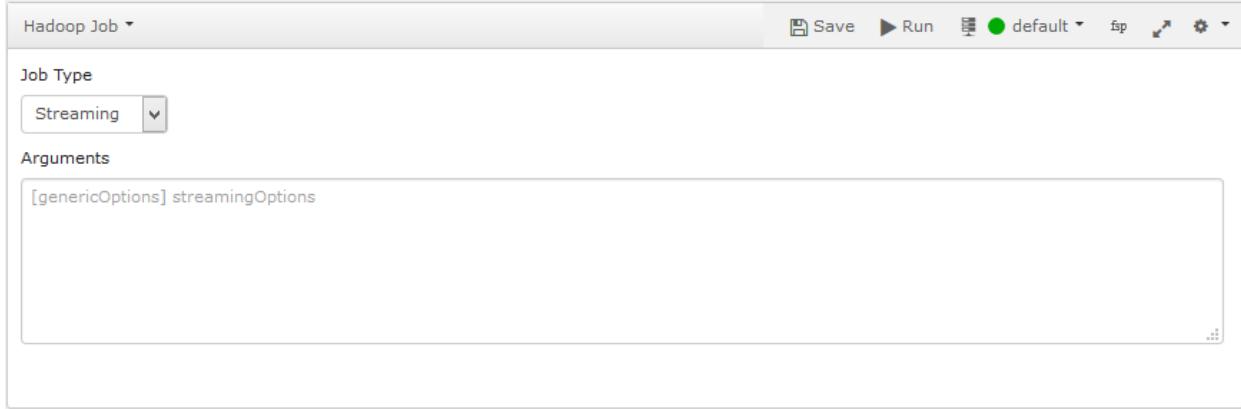
2. **Custom Jar** is selected by default in the **Job Type** drop-down list.
3. Specify the path of the S3 bucket directory that contains the Hadoop jar file in the **Path to Jar File** text field.
4. In the **Arguments** text field, specify the main class, generic options, and other arguments as shown in the figure provided above.
5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

For REST API-related information, see [Submitting a Hadoop Jar Command](#).

Compose a Hadoop Streaming Query

Perform the following steps to compose a Hadoop streaming job query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Hadoop Job** from the **Command Type** drop-down list.
2. Select **Streaming** from the **Job Type** drop-down list. The query composer for Hadoop streaming is as shown in the following figure.

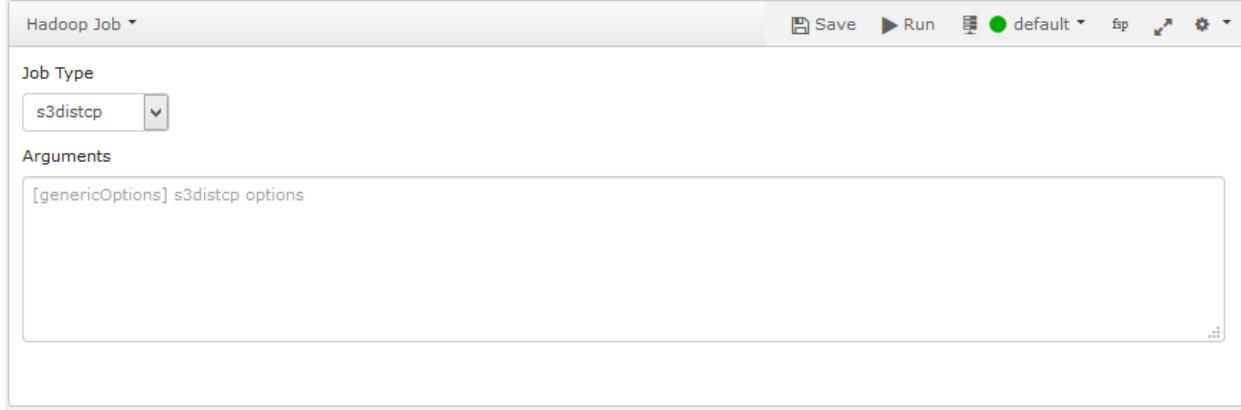


3. In the **Arguments** text field, specify the streaming and generic options.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

Compose a Hadoop S3DistCp Command

Perform the following steps to compose a Hadoop S3DistCp job query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Hadoop Job** from the **Command Type** drop-down list.
2. Select **s3distcp** from the **Job Type** drop-down list. The query composer for Hadoop streaming is as shown in the following figure.



3. In the **Arguments** text field, specify the generic and S3DistCp options.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

For API-related information, see [Submitting a Hadoop S3DistCp Command](#).

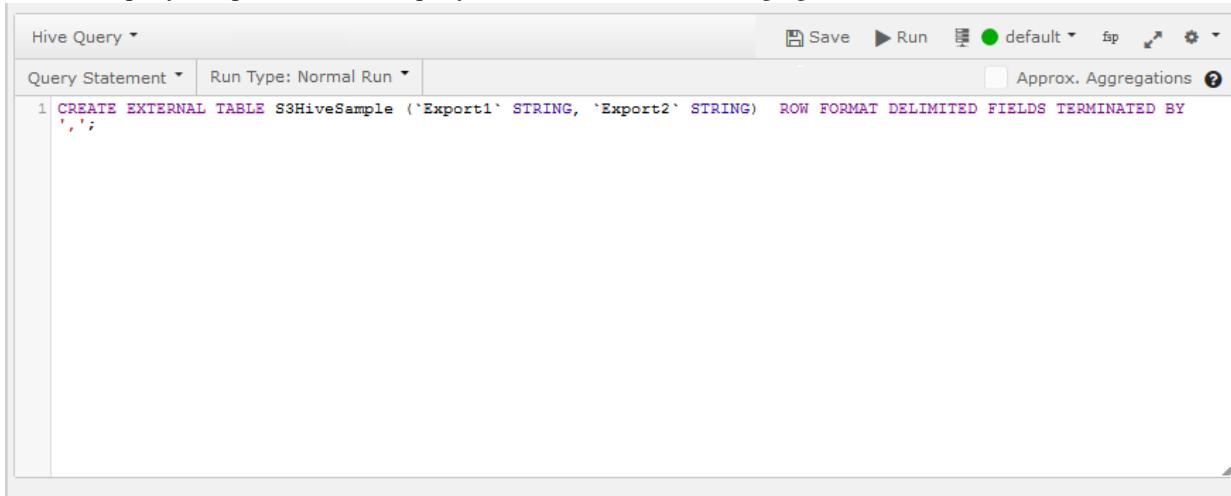
Composing a Hive Query

You can compose a hive query using the query composer available in the **Analyze** page. See *Hive in Qubole* for more information.

Note: Hadoop-1, Hadoop-2, and Presto clusters support Hive queries. See *Mapping of Cluster and Command Type* for more information.

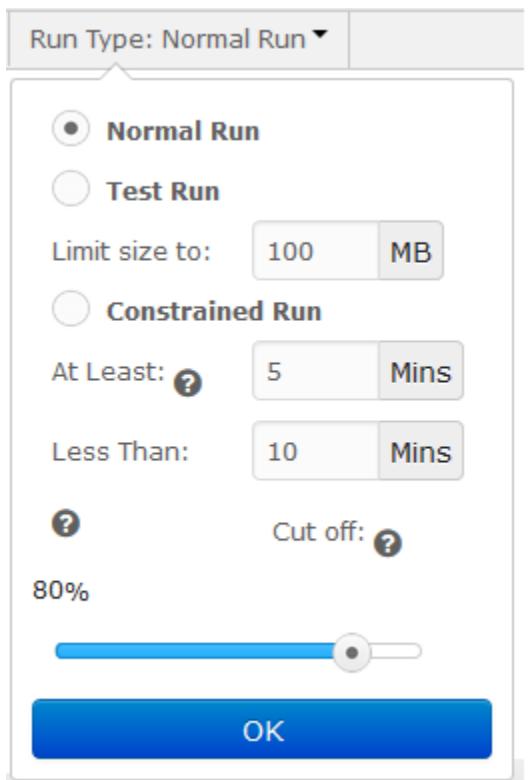
Perform the following steps to compose a Hive query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Hive Query** from the **Command Type** drop-down list.
2. **Query Statement** is selected by default from the drop-down list, which has **Query Path** as the other option. The query composer for a Hive query is as shown in the following figure.



```
1 CREATE EXTERNAL TABLE S3HiveSample ('Export1' STRING, 'Export2' STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

3. In the text field, enter the Hive query. If you select **Query Path**, then specify the S3 bucket directory path that contains the Hive query file. See *Composing a Hive Query by Specifying a Query Path* for more information. The default **Run Type** is **Normal**. **Test Run** and **Constrained Run** are the other run types in the drop-down list, which are illustrated in the following figure.



The primary use of **Test Run** and **Constrained Run** options is during query authoring. Query authoring is often an iterative and error-prone process. When the input datasets are large, it may take minutes or hours for a query to complete and therefore, it takes a long time for you to figure out bugs in expressions and function calls. These options help you run the same query against a small subset of data to get to see some results quickly and this helps to fine-tune Hive queries.

Test Run can be used when you want to read a maximum size (in MB) of data by limiting the dataset size. The default data size to be limited is 100 MB.

Constrained Run can be used when, irrespective of the data, you want the query to run for a time range and this is done by specifying a minimum time (in minutes) and maximum time (in minutes). The default time range is 5 to 10 minutes.

Note: Use the tooltip to know more information on each field.

4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the Job Tracker URL in the **Resources** tab.

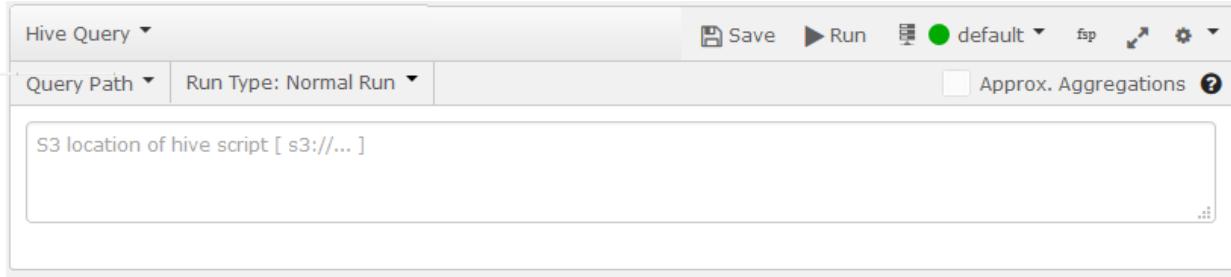
For API-related information, see [Submit a Hive Command](#).

Composing a Hive Query by Specifying a Query Path

Perform the following steps to compose a Hive query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Hive Query** from the **Command Type** drop-down list.

2. Select **Query Path** from the drop-down list that has **Query Statement** selected by default. The query composer for a Hive query path is as shown in the following figure.



3. In the query path text field, specify the S3 bucket directory path that contains the Hive query file. The default **Run Type** is **Normal**. **Test Run** and **Constrained Run** are the other run type in the drop-down list. The two options are explained above.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the Job Tracker URL in the **Resources** tab.

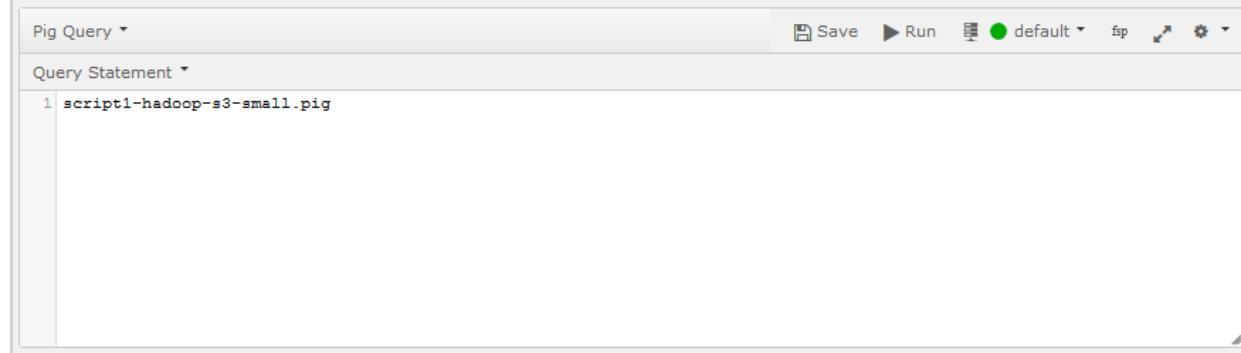
Composing a Pig Query

You can compose a Pig query using the query composer available in the **Analyze** page. See [Running a Pig Job](#) and [Pig in Qubole](#) for more information.

Note: Hadoop-1, Hadoop-2, and Presto clusters support Pig queries. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to compose a Pig query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Pig Query** from the **Command Type** drop-down list.
2. **Query Statement** is selected by default from the drop-down list, which has **Query Path** as the other option. A sample Pig query in the composer is as shown in the following figure.



3. In the text field, enter the Pig query. If you select **Query Path**, then specify the S3 bucket directory path that contains the Pig query file. See [Composing a Pig Query by Specifying a Query Path](#) for more information.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

For REST API-related information, see [Submit a Pig Command](#).

Composing a Pig Query by Specifying a Query Path

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Pig Query** from the **Command Type** drop-down list.
2. Select **Query Path** from the drop-down list that contains **Query Statement** selected by default. A sample query composer for a Pig query path is as shown in the following figure.

3. In the text field, enter the S3 location that contains the pig query path. Enter the script parameters in the **Pig S3 script parameters** in this format, **key1=value1!key2=value2....**
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

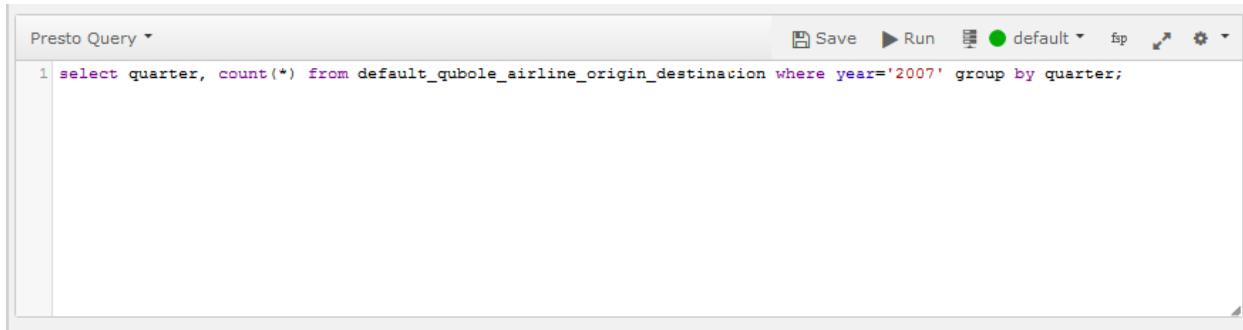
Composing a Presto Query

You can compose a Presto query using the query composer available in the [Analyze](#) page. See [Presto In Qubole](#) for more information.

Note: Presto clusters support Presto queries. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to compose a Presto query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Presto Query** from the **Command Type** drop-down list. A sample Presto query in the composer is as shown in the following figure.



A screenshot of the Qubole Data Service interface showing the 'Presto Query' editor. The query text area contains the following SQL code:

```
1 select quarter, count(*) from default_qubole_airline_origin_destinacion where year='2007' group by quarter;
```

The editor has a toolbar with icons for Save, Run, and other options. The status bar shows 'default' and 'fp'.

2. In the text field, enter the Presto query.
3. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
4. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

For REST API-related information, see [Submit a Presto Command](#).

Composing a Redshift Query

Compose a Redshift query using the query composer available in the **Analyze** page.

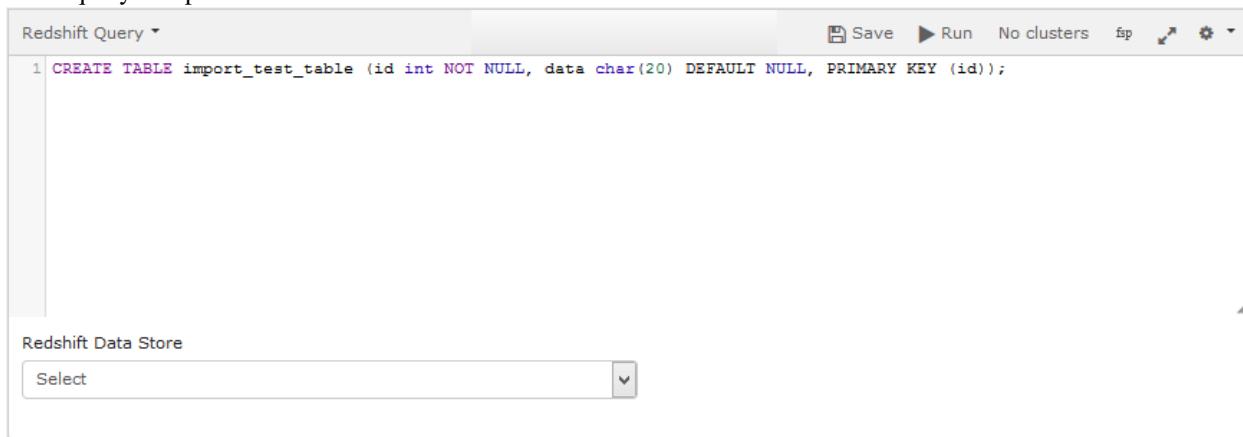
Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support Redshift queries and the queries can run without clusters too. See [Mapping of Cluster and Command Type](#) for more information.

Prerequisites

You must have an existing data store to query it. Create a data store in **Explore** if it does not exist.

Perform the following steps to compose a Redshift query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Redshift Query** from the **Command Type** drop-down list.
2. In the query editor, compose the RedShift query. The following figure shows a sample Redshift query in the query composer.



A screenshot of the Qubole Data Service interface showing the 'Redshift Query' editor. The query text area contains the following SQL code:

```
1 CREATE TABLE import_test_table (id int NOT NULL, data char(20) DEFAULT NULL, PRIMARY KEY (id));
```

The editor has a toolbar with icons for Save, Run, and other options. The status bar shows 'No clusters' and 'fp'.

Below the query editor, there is a section labeled 'Redshift Data Store' with a dropdown menu containing the option 'Select'.

3. Pick a data store from the **Redshift Data Store** drop-down list.

4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

Composing a Refresh Table Query

You can compose a query to refresh Hive tables using the query composer available in the **Analyze** page. See [Refresh Table](#) for more information.

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support refresh table queries and the queries can run without clusters too. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to compose a Refresh Table query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Refresh Table** from the **Command Type** drop-down list. The query composer for **Refresh Table** is as shown in the following figure.

The screenshot shows the 'Refresh Table' query composer interface. At the top, there's a header bar with 'Save', 'Run', 'No clusters', and other icons. Below the header, there are three input fields: 'Hive Database' set to 'default', 'Hive Table Name' set to 'default_qubole_airline_...', and 'Stability Interval (in minutes)' set to '40'. The background of the interface is white, and the text is in a standard sans-serif font.

2. Select a Hive metastore from the **Hive Database** drop-down list.
3. Select the Hive table for which the data must be refreshed from the **Hive Table Name** drop-down list.
4. Set the **Stability Interval** in minutes (numeric value).
5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

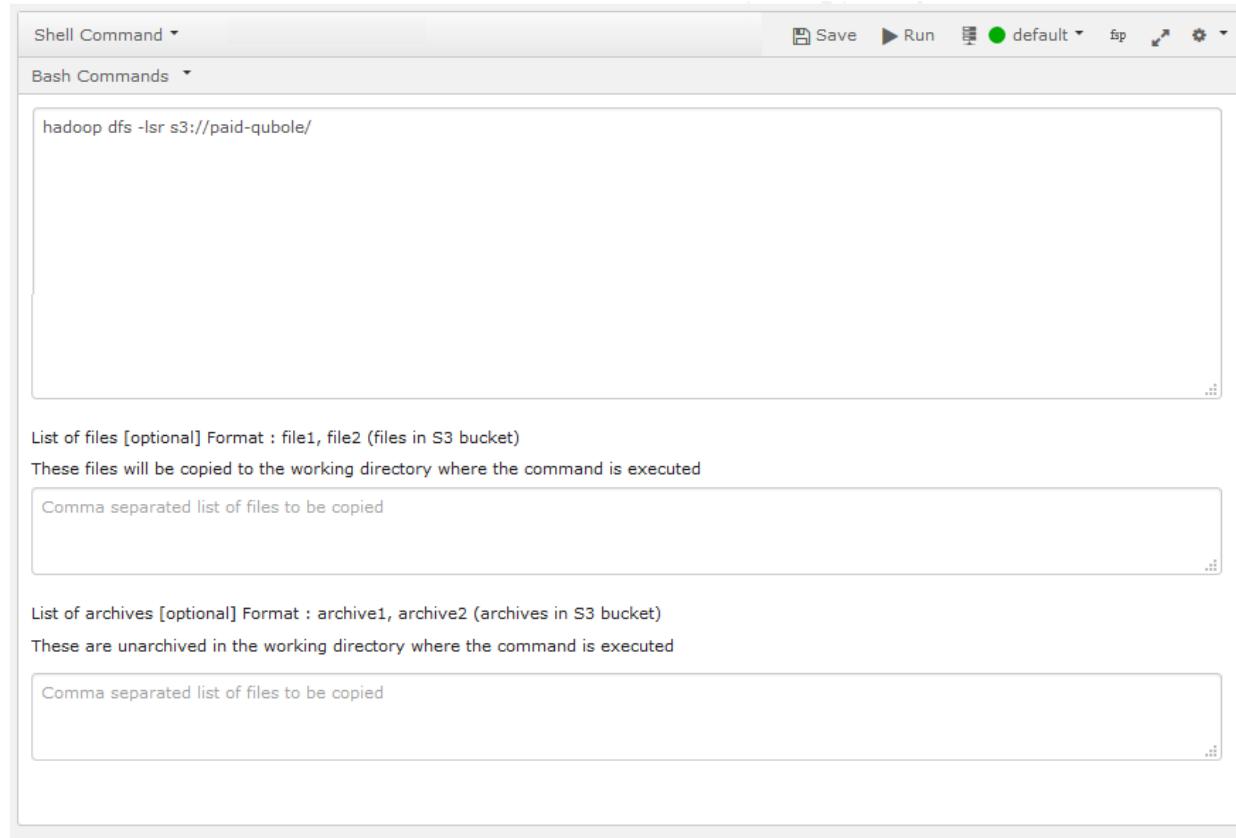
Composing a Shell Command

You can compose a shell command using the query composer available in the **Analyze** page. See [Running a Shell Command](#) for more information.

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support shell commands. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to compose a shell command:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Shell Command** from the **Command Type** drop-down list.
2. **Bash Commands** is selected by default from the drop-down list. **S3 Script Location** is the other option. A sample Shell command in the query composer is as shown in the following figure.



3. In the text field, enter the Shell command. If you select **S3 Script Location**, then specify the S3 script location that contains the shell command script. You can also enter script parameters in the S3 Script Parameters (it is an optional text field).
4. You can list the files separated by a comma in the **List of Files** text field. It is an optional text field and the format is **file1,file2 (files in S3 bucket)**.
5. You can list the archives separated by a comma in the **List of Archives** text field. It is an optional text field and the format is **archive1,archive2 (archives in S3 bucket)**.
6. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
7. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

For REST API-related information, see [Submit a Shell Command](#).

Composing a Spark Command

You can compose a spark command using the query composer available in the **Analyze** page. See [Running Spark Applications](#) and [Spark in Qubole](#) for more information. For REST API-related information, see [Submit a Spark Command](#).

Note: Spark clusters support Spark queries. See [Mapping of Cluster and Command Type](#) for more information.

Spark applications can be composed in:

- Command Line. See [Compose a Spark Application in Command Line](#).

- Python. See [Compose a Spark Application in Python](#).
- Scala. See [Compose a Spark Application in Scala](#).
- SQL. See [Compose a Spark Application in SQL](#).
- 18. See [Compose a Spark Application in R](#).

Note: You can access a Spark job's logs even after the cluster on which it was run is terminated. For more information, see this [blog](#).

Note: You can use the --packages option to add a list of comma-separated maven coordinates of external packages that are used by a Spark application composed in all supported languages. For example, in the **Spark Submit Command Line Options** text field, enter --packages com.package.module_2.10:1.2.3.

Compose a Spark Application in Scala

Perform the following steps to compose a spark query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Spark Command** from the **Command Type** drop-down list.
2. By default, **Scala** is selected. Compose the spark application in scala in the query editor. The query composer with **Spark Command** as the command type is as shown in the following figure.

The screenshot shows the Qubole Data Service Query Composer interface. At the top, there's a toolbar with Save, Run, and other options. Below it, a dropdown menu says "Spark Command". A "Scala" tab is selected, showing the following Scala code:

```

1 import scala.math.random
2 import org.apache.spark._
3 val slices = 6
4 val n = 100000 * slices
5 //spark context is available as sc or spark.
6 val count = sc.parallelize(1 to n, slices).map { i =>
7   val x = random * 2 - 1
8   val y = random * 2 - 1
9   if (x*x + y*y < 1) 1 else 0
10 }.reduce(_ + _)
11 println("Pi is roughly " + 4.0 * count / n)

```

Below the code editor are three large text input fields:

- Spark Submit Command Line Options**: An empty text area.
- Spark Submit Default Command line options**: A text area containing the command line options: `--num-executors 2 --executor-cores 4 --executor-memory 5120M --conf spark.storage.memoryFraction=0.5 --conf spark.yarn.executor.memoryOverhead=1024`.
- Arguments for User Program**: An empty text area.

3. Enter the submit command options in the **Spark Submit Command Line Options** text field to override the default command options in the **Spark Default Submit Command Line Options** text field. This is an optional step.
4. Specify arguments in the **Arguments for User Program**. This is an optional step.
5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the **Spark Application UI URL** in the **Resources** tab.

Compose a Spark Application in Python

Perform the following steps to compose a spark query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Spark Command** from the **Command Type** drop-down list.
2. By default, **Scala** is selected. Select **Python** from the drop-down list. Compose the spark application in python in the the query editor. The query composer with **Spark Command** as the command type is as shown in the following figure.

The screenshot shows the Qubole Data Service interface for composing a Spark application. At the top, there's a navigation bar with 'Save', 'Run', 'default', and other settings. Below it is a 'Spark Command' editor with a Python code snippet for calculating Pi using Monte Carlo simulation. The code uses `random` and `operator` modules, and imports `SparkContext` from `pyspark`. It defines a function `f` that checks if a point (x, y) is within a unit square. The main logic involves parallelizing a range of numbers, mapping them through the function, and reducing the results to count points inside the square. The result is then printed as an approximation of Pi.

Spark Submit Command Line Options

Spark Submit Default Command line options

```
--num-executors 2 --executor-cores 4 --executor-memory 5120M --conf spark.storage.memoryFraction=0.5 --conf
spark.yarn.executor.memoryOverhead=1024
```

Arguments for User Program

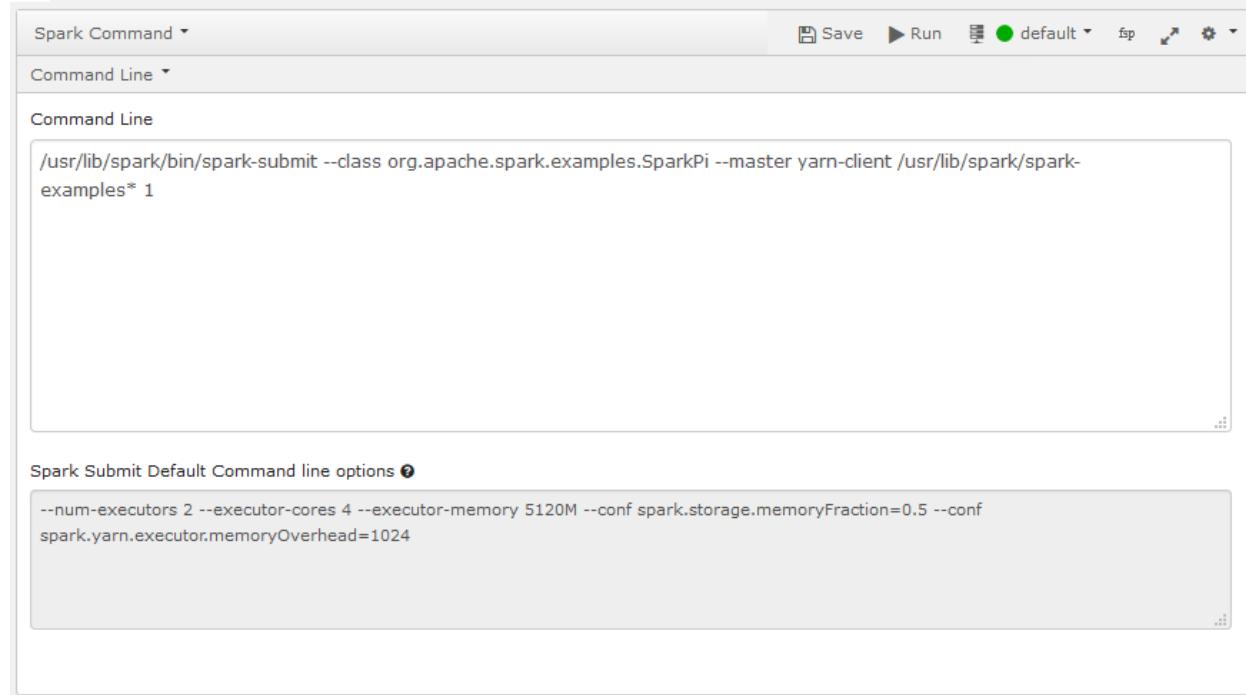
3. Enter the submit command options in the **Spark Submit Command Line Options** text field to override the default command options in the **Spark Default Submit Command Line Options** text field. This is an optional step.
4. Specify arguments in the **Arguments for User Program**. This is an optional step.
5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See *Repo* for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the **Spark Application UI URL** in the **Resources** tab.

Compose a Spark Application in Command Line

Perform the following steps to compose a spark query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Spark Command** from the **Command Type** drop-down list.
2. By default, **Scala** is selected. Select **Command Line** from the drop-down list. Compose the spark application in command line in the the query editor. There are default command options in the **Spark Default Submit**

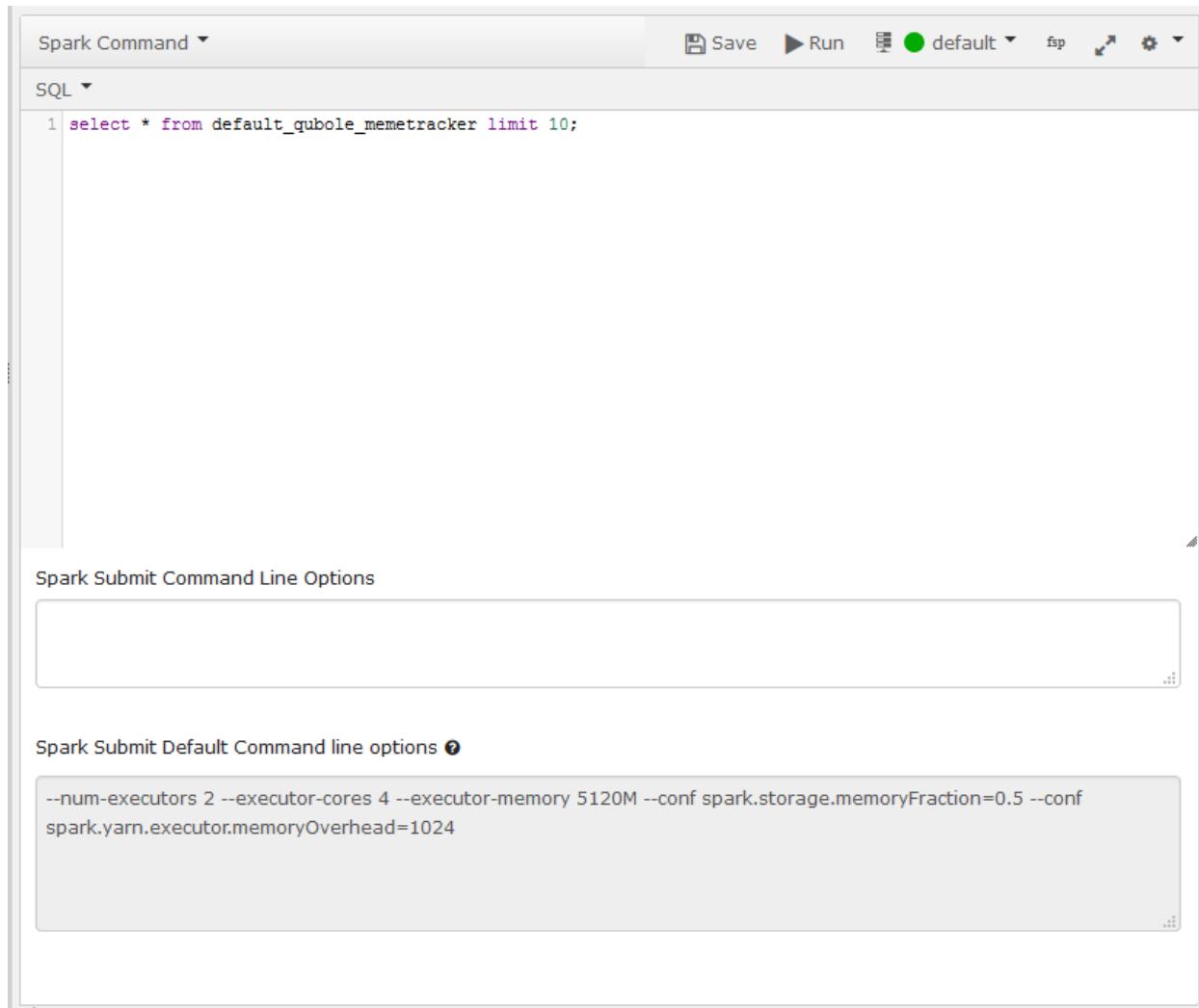
Command Line Options text field that can be overridden by specifying other options. The query composer with **Spark Command** as the command type is as shown in the following figure.



3. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See *Repo* for more information on saving queries.)
4. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the **Spark Application UI URL** in the **Resources** tab.

Compose a Spark Application in SQL

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Spark Command** from the **Command Type** drop-down list.
2. By default, **Scala** is selected. Select **SQL** from the drop-down list. Compose the spark application in SQL in the query editor. The query composer with **Spark Command** as the command type is as shown in the following figure.



3. Enter the submit command options in the **Spark Submit Command Line Options** text field to override the default command options in the **Spark Default Submit Command Line Options** text field. This is an optional step.
4. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
5. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the **Spark Application UI URL** in the **Resources** tab.

Compose a Spark Application in R

Perform the following steps to compose a spark query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Spark Command** from the **Command Type** drop-down list.
2. By default, **Scala** is selected. Select **R** from the drop-down list. Compose the spark application in R in the query editor. The query composer with **Spark Command** as the command type is as shown in the following figure.

The screenshot shows the Qubole Data Service Documentation interface for a Spark command. At the top, there's a toolbar with 'Save', 'Run', and other settings. Below it is a dropdown menu for 'Spark Command' and a dropdown for 'default'. The main area contains an R script:

```

1 library(SparkR)
2 sc <- sparkR.init("yarn-client")
3 sqlContext <- sparkRSQl.init(sc)
4
5 df1 <- jsonFile(sqlContext, "s3n://bucket/dir/data.json")
6 collect(df1)

```

Below the script, there are three sections: 'Spark Submit Command Line Options', 'Spark Submit Default Command line options', and 'Arguments for User Program', each with a text input field.

3. Enter the submit command options in the **Spark Submit Command Line Options** text field to override the default command options in the **Spark Default Submit Command Line Options** text field. This is an optional step.
4. Specify arguments in the **Arguments for User Program**. This is an optional step.
5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab. See the **Spark Application UI URL** in the **Resources** tab.

Composing a Workflow Query

Compose a workflow query when more than one command have to be executed in a sequence (order) using the query composer available in the **Analyze** page. For REST API-related information, see [Submit a Composite Command](#).

Note: Hadoop-1, Hadoop-2, Presto, and Spark clusters support workflow queries. However, the cluster support depends on the type of query that is in a sequence. See [Mapping of Cluster and Command Type](#) for more information.

Perform the following steps to compose a workflow query:

1. Navigate to the [Analyze](#) page and click **Compose**. Select **Workflow** from the **Command Type** drop-down list. The query composer is displayed as shown in the following figure.

2. Click **+ Add Command** to create a new query. After you click it, the query composer is displayed as shown in the following figure.

3. Pick the type of command that you want from the **Select Command Type** drop-down list.
4. Compose the query and click **+ Add Command** to add another query. Repeat steps 2-4 to add the next type of query. Similarly, you can add the required number of queries to be executed in a sequence.

The following figure shows a workflow query composed to execute a data import, hive, and data export queries in the same order.

5. Click **Run** to execute the query. Click **Save** if you want to run the same query later. (See [Repo](#) for more information on saving queries.)
6. The query result is displayed in the **Results** tab and the query logs in the **Logs** tab.

2.2.4 Command Templates

Command Templates is a QDS feature that provides you a template to compose a command/query once and modify a parameter value multiple times or add another parameter along with its value.

If you want to run the same command multiple times with a different set of parameters, instead of rewriting the commands, you can use command templates. A command template contains two variable fields known as **Form Fields** and **Macros**. **Form Fields** are the inputs (parameter/value) that are provided when you run a query. **Macros** are dynamically determined but are not required to be provided as inputs when you run a query.

For example, you can create a command template for the following command.

```
SELECT CouponType FROM default_qubole_airline_origin_destination WHERE DestStateName = 'New York';
```

The same command can be run for other destination states by changing the value for **DestStateName**.

The following topics explain about how to view, create, and edit command templates:

Viewing a Command Template

To see a command template, click **Templates** from the QDS user interface sidebar.

The command templates are listed if the current account contains templates created by all users of the current account. **Templates** appears blank when no template is created in that account.

The following figure shows a list of command templates for a specific account.

The screenshot shows the QDS user interface. On the left, there is a sidebar with a 'Templates' tab selected. Below it, a search bar contains the query 'user_id>All & type>All'. A 'Filter' button is also present. The main area displays a table of command templates with columns: Name, Command, Type, and Recent Runs. The table lists several entries, including 'temp_pig', 'test1', 'create_template2', 'test_template', 'Pig Command Test', 'PigCommand', 'prest_tmp', 'smoke_test_temp', and 'generic_usage_report'. The 'generic_usage_report' row is currently selected, indicated by a blue highlight. On the right side of the screen, a detailed view of the selected template is shown. It includes the 'Template Id 527' and 'Created by qubole@qubole.com' information. Below this, there is a section titled 'Run Command' with a dropdown for 'Cluster Label' set to 'default'. There is also a checkbox for 'Send me an email on run completion' and two buttons: 'Run' and 'Confirm Query'. At the bottom, there is a link labeled 'Details'.

Click a command template to see its details. The right-side of the page displays the command template run details by default. Click **Details** to see the command template details such as **Input Variables** (Form Fields) and **Macros** if any of them are set. The following figure illustrates a template's details.

Template Id 527

Created by qubole@qubole.com

[Run Command](#)[Details](#)

Command i

Hive Query

Hive Statement

```
1 show tables;
```

Macros i

Input Variables i

If you want to see a specific command template and you know its name/ID, then you can view it by using the filter. See *Filtering a Command Template* for more information.

Click the **edit icon**  for editing command template.

Click the **cross-mark icon**  for removing a command template.

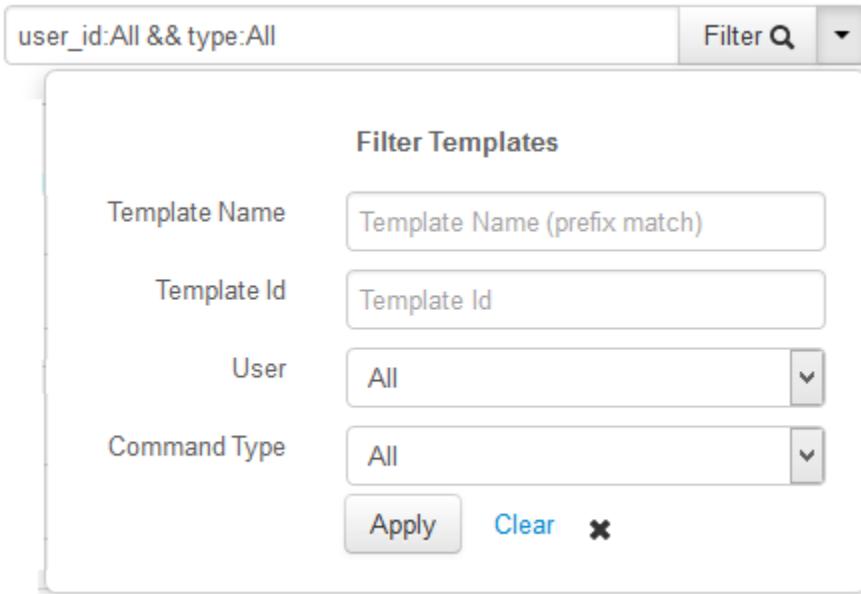
Click the **clone icon**  for cloning a command template.

While cloning a command template, you can retain other details but you must rename the template.

Click the **permalink icon**  to see the command template's permalink.

Filtering a Command Template

In Templates, click **Filter** to look for a specific command template if you know its name/ID. The **Filter Templates** dialog is displayed as shown in the following figure.



The screenshot shows a 'Filter Templates' dialog box. At the top, there is a search bar containing the text 'user_id>All && type>All'. To the right of the search bar is a 'Filter' button with a magnifying glass icon and a dropdown arrow. Below the search bar, the title 'Filter Templates' is centered. There are four filter fields: 'Template Name' (text input field), 'Template Id' (text input field), 'User' (dropdown menu with 'All' selected), and 'Command Type' (dropdown menu with 'All' selected). At the bottom of the dialog are three buttons: 'Apply' (blue), 'Clear' (blue), and a close 'X' button.

Enter the name in the **Template Name** text field and template ID in the **Template ID** text field. Select a specific user from the **User** drop-down list if you do not want the default (All) users option. Select a required command type from the **Command Type** drop-down list if you do not want to see all command types.

Click **Apply** to view the details of the specified command template. Click **Clear** to reenter the values in any **Filter Templates** text field.

Creating a Command Template

You can create a new command template using the QDS user interface.

Perform the following steps to create a new template:

1. Navigate to [Templates](#) and click **New Template**.

Create Command Template is as shown in the following figure.

Create Command Template

Template Name

Command i

Hive Query ▾

Hive Statement ▾

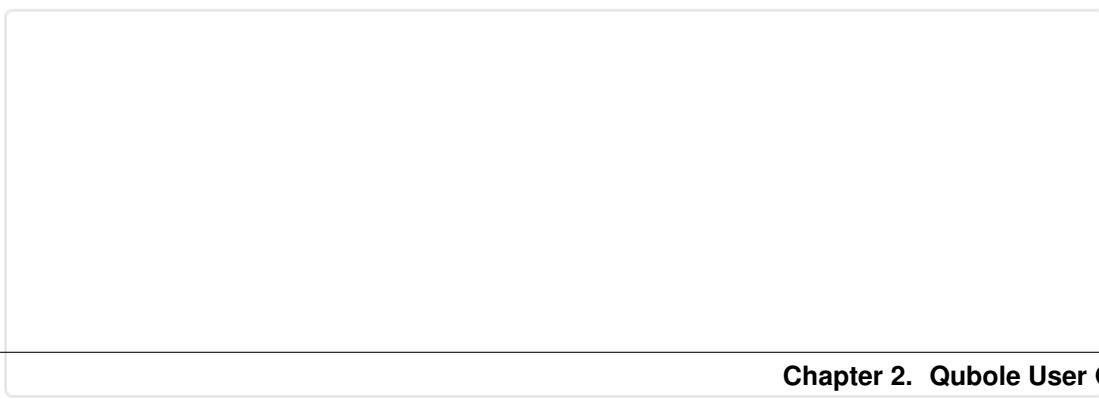
1



Find Variables

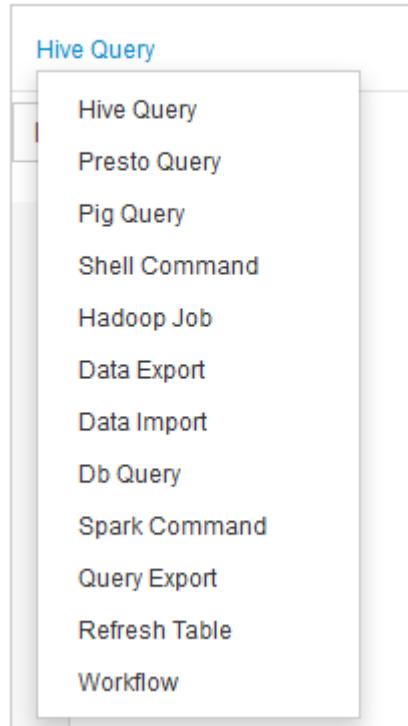
Form Fields i

Click '+' to add



2. Enter a name in the **Template** text field. A system-generated ID is assigned to a created template after it is saved.
3. In the **Command** text field, select a query type from the drop-down list that contains **Hive Query** selected by default. The following types of queries are supported by the command template.

Command



See [Analyze](#) for more information on composing different types of queries.

While entering values for variables, modify the query as shown in the following example:

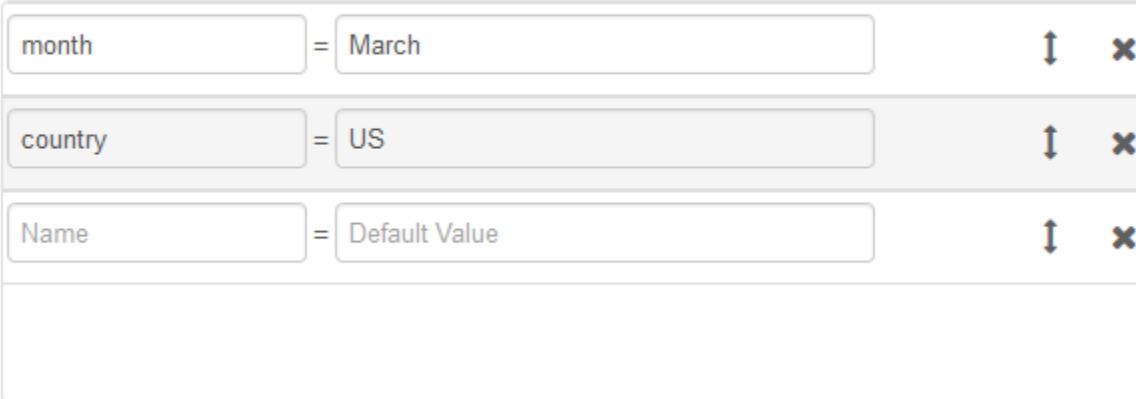
- Consider modifying a sample query as `select col1, col2 from table1 where col3="var"` and set var to hello world for the final query to be `select col1, col2 from table1 where col3="hello world".`

If the query statement is in an S3 location, select **S3 Script Location** from the drop-down list that contains **<CommandType> Statement** selected by default. Enter the S3 location that contains the script. Qubole now allows setting an S3 script location for Presto queries.

4. Set the parameters in **Form Fields**. Click **+** to add parameters for the query. For example, if you compose a query for a table that contains *Month* and *Country* as columns to get a result limited to the two columns, then **Form Fields** can be as shown in the following figure.

Form Fields

Click '+' to add



month	=	March	 
country	=	US	 
Name	=	Default Value	 

Enter a parameter's name in the first text field and the value in the second text field. Click + to add more parameters for a query as necessary. While entering variables in template screen, escape the double quotes. That is, escape the double-quote that must be present in a string and wrap the entire string in double-quotes. For example, " "hello world" " gets substituted as select col1, col2 from table1 where col3= "hello world".

Click the **cross-mark icon**  for removing a form field.

Click the **double-arrow icon**  to change the order of a form field by dragging and dropping it.

5. Adding **Macros** is optional. To use macros in the query, click the **+Add Macro** button available in the **Macros** field. Else, proceed to the next step. After you click the **+Add Macro** button, the macros are displayed as shown in the following figure.

Macros



Variable	=	Expression	
----------	---	------------	---

+Add Macro **Validate Macros**

Enter the variable and expression and click **Validate Macros** for validating it. See [Macros in Scheduler](#) for more information. Click **+Add Macro** to add another macro. Else, proceed to the next step.

6. Click **Submit** to add a new command template to the existing list if any.

Editing a Command Template

From the list of command templates available in [Templates](#), you can modify an existing command template.

Perform the following steps to edit a command template:

1. Select a template that you want to edit from the list of command templates. The following figure shows a template that is selected from the templates' list.

The screenshot shows the Qubole Command Template Editor. On the left, a list of templates is displayed with columns for Name, Command, and Type. One template, "PigCommand", is selected and highlighted in blue. On the right, a detailed view of this template is shown. It includes fields for "Run Command" (with entries for test, test1, test2, and test3), "Cluster Label" (set to "default"), and checkboxes for "Send me an email on run completion". At the bottom, there are "Run" and "Confirm Query" buttons.

Name	Command	Type
temp_pig	tedy	PigCommand
test1	show create \$what\$;	HiveCommand
create_template2	show \$what\$;	HiveCommand
test_template	select * from \$\$;	HiveCommand
Pig Command Test	test query template	PigCommand
PigCommand	select * from \$test\$ limit \$test\$;	PigCommand
prest_tmp	select * from \$test\$;	PrestoCommand
smoke_test_temp	show tables;	HiveCommand
generic_usage_report	select a.account_id,d.name,c.tag, ceil(sum((case amazon_instance_type when 't1.micro' and account_id IN (\$account_ids\$) group by a.account_id,c.tag,d.name order by acuh	DbTapQueryCommand

2. Click the **edit icon** for editing it.

Edit Command Template is displayed as shown in the following figure.

Edit Command Template

Template Name

PigCommand

Command i

Pig Query▼

Pig Statement ▾

```
1 select * from $test$ limit $test3$;
```

Find Variables

Form Fields i

Click '+' to add



test	=	'asda'	↔	×
test1	=	1	↔	×
test2	=	2	↔	×
test3	=	test1-test2	↔	×

Macros i

Click '+' to add



3. Edit the template name if you want to change it.
4. Edit the query statement if it is required.
5. Change the **Form Field** values and add a new parameter by clicking + sign if it is required. See [Creating a Command Template](#) for more information.
6. Adding **Macros** is optional. To use macros in the query, click the **+Add Macro** button available in the **Macros** field. See [Creating a Command Template](#) for more information.
7. Click **Submit** to save the changes.

2.2.5 Control Panel

About Control Panel

Control Panel is the user interface (UI) that serves as the administration console to manage cluster settings, hive-bootstraps, sessions, user accounts, user roles, access permissions, and payment/subscription.

Navigating to **Control Panel** opens the page with **Clusters** as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)					
Search : Enter Search Text							
Id		Labels		Nodes	Up Time	Resources	Action
●	4633 ✓	default		0		Cluster Usage Report	
●	4634	presto		0		Cluster Usage Report	
●	4635	hadoop2		0		Cluster Usage Report	
●	4636	spark		0		Cluster Usage Report	

The settings and profiles that can be managed using the Control Panel are described in the following sections:

- [Managing Account Settings](#)
- [Managing Clusters](#)
- [Managing Groups](#)
- [Managing Hive Bootstrap](#)
- [Managing My Account](#)
- [Managing Profile](#)
- [Managing Roles](#)
- [Managing Sessions](#)
- [Managing Subscription and Payment](#)
- [Managing Users](#)

Managing Account Settings

In the [Account Settings](#) page, you can set **Account Details**, **Storage Settings** and **Compute Settings**. Qubole allows you to authorize Amazon Web Service (AWS) using:

- IAM Keys to access AWS resources. See [Authorizing AWS using IAM Keys](#) for more information.
- IAM Roles to access AWS resources. See [Authorizing AWS using IAM Roles](#) for more information. To know more about IAM roles, click [here](#).

About IAM Roles in QDS

- IAM role is an AWS recommended practice and provides a very secure experience as compared to IAM Keys for accessing AWS resources.
- Cross-account IAM roles can be configured only at the Qubole account level and used for:
 - All Storage access
 - All Compute access. This includes all clusters associated with the account.
- Qubole Data service currently supports the AWS IAM-role based authentication on only Hadoop-1, Hadoop-2, Presto, and Spark clusters, and the following type of commands with IAM roles:
 - Hive commands
 - Hadoop commands on Hadoop-1 and Hadoop-2 clusters:
 - * Custom JAR
 - * Streaming
 - Pig on Hadoop-1 and Hadoop-2 clusters
 - Presto commands
 - Shell commands
 - Spark commands

Authorizing AWS using IAM Keys The options that can be set in **Account Details** are as shown in the following figure.

The screenshot shows the 'Account Settings' page. It has three main sections: 'Account Details', 'Notifications', and 'Access Mode (Keys / IAM Roles)'. In 'Account Details', there are fields for 'Account Name' (qubole) and 'Domain Name Allowed to Sign In/Up' (xyz.com). In 'Notifications', there is a field for 'Email List for Account Updates' (qubole@qubole.com) and a 'Command Timeout' field set to 'seconds'. In 'Access Mode (Keys / IAM Roles)', the 'Type' is set to 'IAM Keys' (selected by default). At the bottom are 'Save' and 'Reset' buttons.

- **Account Details:**

- **Account name** - You can set name in the text field.
- **Domain Name Allowed to Sign In/Up** - You can select the domains that you want to sign up and sign in. You can set multiple domains separated by a comma.

- **Notifications:**

- You can add emails that receive notifications about account changes in the **Email List for Account Updates** text field.
- **Command Timeout** - It is set in seconds for the queries that you run in the query composer of the Analyze page.

- **Access Mode (Keys / IAM Roles)** - **IAM Keys** is selected by default.

See [What policy should I use for Qubole to use my IAM credentials?](#) for the set of permissions to be given.

Storage Settings with IAM Keys When you create an account with QDS, you get two options:

- Use default storage settings (managed by Qubole)
- Use own credentials

Note: Once you set own credentials, you cannot revert to the default storage.

The options in **Storage Settings** are as shown in the following figure with the default storage.

The screenshot shows the 'Storage Settings' configuration page. It includes fields for 'Access Key' (set to 'AKIATMTATMTATMTA') and 'AWS Secret Key' (set to 'GXXCMs6/as6/as6/as6/as6/as6/as6/as6/as6/asJ6r'). The 'Default Location (for the created data)' is set to 'data.com/user_hu_481'. The 'S3 Cache Size' is set to '25 GB'. There are two checkboxes: 'Allow download from S3' (unchecked) and 'Allow upload to S3' (unchecked). At the bottom are 'Save' and 'Reset' buttons.

Qubole provides storage for a 15-day trial period. To use own storage credentials, click **Configure**.

The options in **Storage Settings** are as shown in the following figure when IAM Key is the access mode.

The screenshot shows the 'Storage Settings' configuration page with IAM Key configuration. It includes fields for 'AWS Access Key' (set to 'ABCDEFGHIJKLMNPQRST') and 'AWS Secret Key' (set to '*****' with a link to 'Edit secret Key'). The 'Default Location (for any data created)' is set to 'data.com/data'. The 'S3 Cache Size' is set to '25 GB'. There are two checkboxes: 'Allow download from S3' (checked) and 'Allow upload to S3' (checked). At the bottom are 'Save' and 'Reset' buttons.

You can modify **AWS Access Key** (must have 20 characters) and **AWS Secret Key** (must have 40 characters). Click **Edit Secret Key** to add a different key. An empty text field is displayed. After adding the secret key, click **Hide Secret Key** to hide it.

You can change the **Default Location (for any data created)** and change the **S3 Cache Size** in GB.

Select **Allow download from S3** to enable downloading files from **My Amazon S3** in the [Explore](#) page.

Select **Allow upload to S3** to enable uploading files to **My Amazon S3** in the [Explore](#) page.

Click **Save** after making changes. Click **Reset** to change any setting.

See [Manage Roles](#) for information on how permissions to access features are granted and restricted.

Compute Settings with IAM Keys The options in **Compute Settings** are as shown in the following figure.

When you create an account with QDS, you get two options:

- Use default compute settings (managed by Qubole)
- Use own credentials

The **Default Compute Settings** is as shown in the following figure.

The screenshot shows a web-based configuration interface for 'Default Compute Settings'. At the top, there's a 'Configure' button. Below it, four input fields are displayed: 'AWS Access Key' (with placeholder 'e.g. GH56GHRD871AFGHYFGA'), 'AWS Secret Key' (with placeholder 'e.g. fngFHgHFFGF6754gvhjf564'), 'Push AWS keys to all clusters' (with a checked checkbox), and 'Security Group' (with an empty input field). At the bottom are 'Save' and 'Reset' buttons.

Qubole provides compute settings for a 15-day trial period. To use own compute credentials, click **Configure**. The **Compute Settings** options are displayed as shown in the following figure.

This screenshot shows the 'Compute Settings' configuration page. It includes fields for 'AWS Access Key' (placeholder 'e.g. GH56GHRD871AFGHYFGA'), 'AWS Secret Key' (placeholder 'e.g. fngFHgHFFGF6754gvhjf564'), 'Push AWS keys to all clusters' (checkbox checked), and 'Security Group' (empty input field). Below these are 'Save' and 'Reset' buttons.

Note: Once you set own credentials, you cannot revert to the default compute settings.

Add **AWS Access Key** (must have 20 characters) and **AWS Secret Key** (must have 40 characters).

You can modify the keys after you save them. Click **Edit Secret Key** to add a different key. An empty text field is displayed. After adding the secret key, click **Hide Secret Key** to hide it.

Push AWS keys to all clusters is selected by default. Unselect this option if you do not want to use it.

Security Group is not set by default. When an account security group is set, it is attached to all clusters by default unless it is overridden in the cluster settings.

Click **Save** after making changes. Click **Reset** to change any setting.

Authorizing AWS using IAM Roles The options that can be set in **Account Details** are as shown in the following figure.

The screenshot shows the 'Account Settings' page with the following configuration:

- Account Details:**
 - Account Name: qubole
 - Domain Name Allowed to Sign In/Up: xyz.com
- Notifications:**
 - Email List for Account Updates: qubole@qubole.com
 - Command Timeout: seconds
- Access Mode (Keys / IAM Roles):**
 - Type: IAM Keys (selected)

At the bottom are 'Save' and 'Reset' buttons.

The options are as follows:

- **Account Details:**
 - **Account name** - You can set name in the text field.
 - **Domain Name Allowed to Sign In/Up** - You can select the domains that you want to sign up and sign in. You can set multiple domains separated by a comma.
- **Notifications:**
 - You can add emails that receive notifications about account changes in the **Email List for Account Updates** text field.
 - **Command Timeout** - It is set in seconds for the queries that you run in the query composer of the Analyze page.

Access Mode

- **Access Mode (Keys / IAM Roles)** - To configure an IAM role, select **IAM Role**. Once you select it, the IAM role settings are displayed as shown in the following figure.

The screenshot shows the 'Account Settings' page under the 'Account Details' section. It includes fields for 'Account Name' (qubole), 'Domain Name Allowed to Sign In/Up' (xyz.com), 'Email List for Account Updates' (qubole@qubole.com), 'Command Timeout' (seconds), and 'Access Mode (Keys / IAM Roles)'. Under 'Access Mode', 'Type' is set to 'IAM Role' (selected radio button). Other fields include 'Qubole AWS Account ID' (8052460), 'External ID' (GUYDCYTDNE), 'Role ARN' (Role ARN placeholder), and 'Default Location(for any data created)' (data.com/data). At the bottom are 'Save' and 'Reset' buttons.

Once, an IAM role is configured, the account uses role credentials instead of access keys to interact with AWS resources.

The following text fields are associated with an IAM role:

- **Qubole AWS Account ID** - It is a Qubole AWS account ID as shown in the above figure. It is used while creating an IAM role using an AWS console.
- **External ID** - It is a unique ID that is generated when a new account is created. It is used while creating an IAM role using an AWS console.
- The **Role ARN** - It is a text field in which the role ARN value is filled. [Creating a Cross-account IAM Role for QDS](#) describes how to create an IAM role during which a Role ARN is generated. Enter the value in **Role ARN** text field.
- **Default location (for any data created)** - It is a location where logs and output data, and so on are stored. Change it if you want a different default location.

Note: If you use a Qubole account created before August 20, 2015, drop an email to solutions@qubole.com to get an early access to the IAM Role feature.

Click **Save** after making changes. Click **Reset** to change any setting.

Storage Settings with IAM Roles The options in **Storage Settings** are as shown in the following figure when IAM Role is the access mode.

The screenshot shows a configuration panel titled "Storage Settings". It includes fields for "S3 Cache Size" (set to 25 GB), "Allow download from S3" (checked), "Allow upload to S3" (checked), and buttons for "Save" and "Reset".

You can change the **S3 Cache Size** in GB.

Select **Allow download from S3** to enable downloading files from **My Amazon S3** in the [Explore](#) page.

Select **Allow upload to S3** to enable uploading files to **My Amazon S3** in the [Explore](#) page.

Click **Save** after making changes. Click **Reset** to change any setting.

See [Manage Roles](#) for information on how permissions to access features are granted and restricted.

Compute Settings with IAM Roles The options in **Compute Settings** are as shown in the following figure.

Compute Settings

The screenshot shows a configuration panel titled "Compute Settings". It includes a "Security group" dropdown, a "Save" button, and a "Reset" button.

Security Group is not set by default. When an account security group is set, it is attached to all clusters by default unless it is overridden in the cluster settings.

Click **Save** after making changes. Click **Reset** to change any setting.

Creating a Cross-account IAM Role for QDS

Qubole allows you to configure a cross-account IAM role as an alternative to access keys to interact with AWS resources.

Note: If you use a Qubole account created before August 20, 2015, drop an email to solutions@qubole.com to get an early access to the IAM Role feature.

Perform the following steps to create an IAM role and set required permissions for configuring an IAM role on QDS:

1. Sign in to the **AWS Management Console** and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Enter the values of Qubole AWS Account ID and External ID as available in the Qubole account settings to create an IAM role. See [Managing Account Settings](#) for more information. Note down the **Role ARN** value that you need to enter in Qubole account settings tab.

Note: Creating an instance profile, and setting IAM permissions and trust policy for the newly-created IAM role are mandatory and the steps 4, 5, and 6 explain how to create/set them.

3. In the created IAM role, create an instance profile. An instance profile must meet the following requirements:

- The instance-profile contains the cross-account role-ARN.
- The instance-profile has the same name as cross-account IAM role.

For example: If The role-ARN is **arn:aws:iam::xxxxxxxxxxxx:role/3rdParty-Role-For-Qubole**, the instance-profile-ARN containing the above role-ARN is **arn:aws:iam::xxxxxxxxxxxx:instance-profile/3rdParty-Role-For-Qubole**.

This instance-profile must be specified while granting the *iam:PassRole* permissions that is mentioned in the next step.

4. Set IAM permissions for the newly created IAM role. See [What policy should I use for Qubole to use my IAM credentials?](#) for the set of permissions to be given to a cross-account IAM role. In addition to the above set of permissions, the following permissions must be granted to the new IAM role:

```
{
    "Effect": "Allow",
    "Action": "iam:GetInstanceProfile",
    "Resource": "<arn-of-your-instance-profile>"
},
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "<arn-of-your-cross-account-IAM-role>"
}
```

5. Append the following settings to the trust policy for the new IAM role.

```
{
    "Sid": "",
    "Effect": "Allow",
    "Principal": {
        "Service": "ec2.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
}
```

6. Enter the Role ARN and default location in [Account Settings](#). See [Access Mode](#) for more information on IAM Roles and Role ARN.

Managing Clusters

[Clusters](#) are pre-configured and are used to run Hadoop, Presto, and Spark queries on the **Compose** command composer and editor of the [Analyze](#) page.

It brings up dedicated Hadoop/Hive clusters for your account when required. Qubole appropriately resizes these clusters depending on the query workload submitted by the users of an account. The cluster lifespan is associated with the notion of sessions. By default, a session lasts for 2 hours. You can create new sessions, terminate old sessions or extend old sessions beyond 2 hours through **Sessions**. A dedicated Hadoop/Hive cluster is maintained for an account by Qubole as long as there is an active session for a user account. An active session can be deactivated and the inactive session can be activated again within the session duration.

The **Cluster** tab shows the current status of the cluster, running nodes, pending nodes (requested from aws), up time of the cluster and link to the master node (Job Tracker for Hadoop, Resource Manager for Hadoop2, Presto Master for Presto clusters) for the running cluster. Qubole intends to put detailed cluster usage charts under this tab.

Once you move from default storage (managed by Qubole) to own storage type, you must change the compute access and secret keys. If you want to modify any cluster configuration, edit it. [Qubole Clusters](#) provides more information.

This section mainly describes editing the following cluster configuration:

- [*Modifying Cluster Settings*](#)
- [*Modifying Cluster Composition*](#)
- [*Modifying EC2 Settings*](#)
- [*Modifying Hadoop Cluster Settings*](#)
- [*Modifying Security Settings*](#)
- [*Setting Cluster as Default*](#)

Modifying Clusters Configuration



To edit a cluster, click the edit icon available in the **Action** column.

Cluster settings are displayed as shown in the following figure.

Cluster > Edit Cluster (5511)

Cluster Information					
Cluster Settings					
Cluster Labels	<input type="text" value="hadoop1"/>				
Cluster Type	<input type="button" value="Hadoop"/>				
Minimum Slave Count	<input type="text" value="2"/>				
Maximum Slave Count	<input type="text" value="2"/>				
Master Node Type	<input type="button" value="m1.large - 2 cores, 7.5GiB m"/>				
Slave Node Type	<input type="button" value="m1.xlarge - 4 cores, 15GiB m"/>				
Node Bootstrap File	<input type="text" value="s3://data.com/qubole/scripts/hadoop/node_bootstrap.sh"/>				
Other Settings	<input type="checkbox"/> Disable Automatic Cluster Termination <small>?</small> <input type="checkbox"/> Enable Ganglia Monitoring <small>?</small>				
Cluster Composition					
Autoscaling Node Purchasing Option	<input type="button" value="Spot Instance"/>				
Use Qubole Placement Policy	<input checked="" type="checkbox"/> <small>?</small>				
Volatile Spot Instance Settings					
Fallback to on demand	<input checked="" type="checkbox"/> <small>?</small>				
Maximum Bid Price	<input type="text" value="100.0"/> % (<small>Pricing: m1.xlarge : \$0.350/hour</small>) EC2 Pricing				
Request Timeout	<input type="text" value="10"/> minutes				
Spot Instances Percentage	<input type="text" value="50"/> %				
Stable Spot Instance Settings					
Use Stable Spot Nodes	<input type="checkbox"/> <small>?</small>				
EC2 Settings <small>?</small>					
Same As Default Compute	<input checked="" type="checkbox"/>				
Compute Access Key	<input type="text" value="ABCDEFGHIJKLMNPQRSTUVWXYZ"/>				
Compute Secret Key	<input type="text" value="*****"/> <small>Edit Secret Key</small>				
Region	<input type="button" value="us-east-1"/>				
Aws Availability Zone	<input type="button" value="No Preference"/> <small>?</small> <input type="button" value=""/>				
VPC	<input type="button" value="None"/> (optional) <small>?</small> <input type="button" value=""/>				
Subnet	<input type="button" value=""/> (optional)				
Custom EC2 Tags	<table border="1"> <thead> <tr> <th>Custom Tag</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td><input type="text" value="Qubole"/></td> <td><input type="text" value="qbol_acc1"/></td> </tr> </tbody> </table> <input type="button" value="+ Add EC2 Tags"/>	Custom Tag	Value	<input type="text" value="Qubole"/>	<input type="text" value="qbol_acc1"/>
Custom Tag	Value				
<input type="text" value="Qubole"/>	<input type="text" value="qbol_acc1"/>				
Hadoop Cluster Settings					
Override Hadoop Configuration Variables	<input type="text"/>				
Recommended Configuration	<pre>mapred.map.child.java.opts=-server -Xmx1024m -Djava.net.preferIPv4Stack=true mapred.reduce.child.java.opts=-server -Xmx1024m -Djava.net.preferIPv4Stack=true mapred.child.java.opts=-server -Xmx1024m -Djava.net.preferIPv4Stack=true mapred.tasktracker.map.tasks.maximum=4 mapred.tasktracker.reduce.tasks.maximum=4</pre>				
2.2. Features <small>?</small> <small>?</small>					
Info: if you are configuring fair scheduler, make sure to add <code>fairSharePreemptionTimeout</code> property. Example: <code><fairSharePreemptionTimeout></fairSharePreemptionTimeout></code>					
Default Fair Scheduler Pool <input type="text"/>					

Edit the settings that you want to modify. The **Edit Cluster** page contains configurations classified as:

1. **Cluster Settings.** See [Modifying Cluster Settings](#).
2. **Cluster Composition.** See [Modifying Cluster Composition](#).
3. **EC2 Settings.** See [Modifying EC2 Settings](#).
4. **Hadoop Cluster Settings.** See [Modifying Hadoop Cluster Settings](#).
5. **Security Settings.** See [Modifying Security Settings](#).

Note: Use the tooltip  to know more information on each field or checkbox.

Modifying Cluster Settings

In **Cluster Settings**, you can modify:

- **Cluster Labels:** A cluster can have one or more labels separated by a comma. You can set/label a cluster as default by selecting the default option.
- **Cluster Type:** It shows the type of cluster. By default, **Hadoop** is the default cluster type on QDS. **Hadoop2**, **Presto**, and **Spark** are the other cluster types.
- **Minimum Slave Count:** Set the minimum number of slave nodes if you want to change it from the default 2.
- **Maximum Slave Count:** Set the maximum number of slave nodes if you want to change it from the default 2.
- **Master Node Type:** Set the master node type by selecting the preferred node type from the drop-down list.
- **Slave Node Type:** Set the slave node type by selecting the preferred node type from the drop-down list. However, if you select the **c3**, **c4**, **m3**, **m4** and **r3** slave node type, you get three additional Elastic Block Storage (EBS) configurations, that is **EBS Volume Count**, **EBS Volume Type** (Magnetic is the default EBS volume type), and **EBS Volume Size** (supported range is 100 - 500 GB).
- **Node Bootstrap File:** Set the node bootstrap file.
- **Other Settings:** To enable/disable **Disable Automatic Cluster Termination** and **Enable Ganglia Monitoring**. By default, the two options are disabled.

[Cluster Basics](#) and [Configuring Clusters](#) provides more information on the cluster and its configuration options.

[How to Push Configuration to a Cluster](#) provides more information on pushing certain configuration to a running cluster.

Modifying Cluster Composition **Cluster Composition** is broadly classified by two types of **Autoscaling Node Purchasing Option**:

- **On-Demand Instance** - This does not have any configuration sub-options.
- **Spot Instance** - By default, **Spot Instance** is selected.

For **Spot Instance**, you can modify:

- **Use Qubole Placement Policy:** It is a checkbox that is enabled by default.
- **Volatile Spot Instance Settings:** You can:
 - Enable **Fallback to on demand**
 - Set **Maximum Bid Price** in percentage if you want to change the default 100%

- Set **Request timeout** if you want to change the default 10 minutes
- Set **Spot Instances Percentage** if you want to change the default 50%
- **Stable Spot Instance Settings:** Select **Use Stable Spot Nodes** to use spot instances for core nodes.

Cluster Composition provides more description about all the options.

Modifying EC2 Settings Amazon Elastic Cloud Compute (EC2) web service offers resizable compute capacity in the cloud. EC2 settings' fields are visible only when the **Compute Type** is managed by you and not Qubole. The following figure shows an example of EC2 settings.

EC2 Settings  

Same As Default Compute	<input type="checkbox"/>						
Compute Access Key	ABCDEFGHIJKLMNPQRSTUVWXYZ						
Compute Secret Key	***** Edit Secret Key						
Region	us-east-1  						
Aws Availability Zone	No Preference  						
VPC	None  (optional) 						
Subnet	 (optional)						
Custom EC2 Tags	<table border="1"> <thead> <tr> <th>Custom Tag</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Qubole</td> <td>qbol_acc1</td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table> <p>+ Add EC2 Tags</p>	Custom Tag	Value	Qubole	qbol_acc1		
Custom Tag	Value						
Qubole	qbol_acc1						
							

Note: If **IAM Roles** is set as the access mode type in account settings, Access and Secret Keys are not displayed in EC2 settings. Instead, the configured **Role ARN** and **External ID** text fields are displayed in **EC2 Settings** (The two fields can be modified in **Control Panel > Account Settings**. Changing the two fields changes compute access and storage access for all clusters of the specific account).

EC2 Settings contains the following options:

- **Same As Default Compute** - Select this checkbox if you want the default compute settings for the cluster.

Note: Once you select **Same as Default Compute**, the **Compute Access Key** and **Compute Secret Key** text fields are not displayed.

- **Compute Access Key** - Enter an access key (along with the secret key) to set AWS Credentials for launching the cluster.
- **Compute Secret Key** - Enter a secret key (along with the access key) to set AWS Credentials for launching the cluster. Click **Edit Secret Key** to add a different key. An empty text field is displayed. Add a different secret key and click **Hide Secret Key**.

- **AWS Region** - Denotes the region from which the nodes are launched. The default region is *us-east-1*. Select a region from the drop-down list to change it.
- **AWS Availability Zone** - By default, the availability zone is set to *No Preference* and the best availability zone based on system health and available capacity is selected. Note that all cluster nodes are in the same availability zone. You can select an availability zone from the drop-down list, in which instances are reserved to benefit from the AWS reserved instances. Click the refresh icon  to refresh the availability zones' list.
- **VPC** - Set this configuration to launch the cluster within an Amazon Virtual Private Cloud (VPC). It is an optional configuration. If VPC is not specified, cluster nodes are placed in a Qubole created Security Group in EC2-Classic or in the *default VPC* for EC2-VPC accounts. By default, **None** is selected. Click the refresh icon  to refresh the VPC list.
- **Subnet** - When you select a VPC, a corresponding subnet gets selected. By default, the text field is blank as no VPC is selected.
- **Custom EC2 Tags** - You can set a custom EC2 tag if you want the instances of a cluster to get that tag on AWS. This tag is useful in billing across teams as you get the AWS cost per tag, which helps to calculate the AWS costs of different teams in a company.

Add a tag in **Custom Tag** and enter a value for the tag in the corresponding text field as shown in the above figure. Tags and values must have alphanumeric characters and can contain only these special characters: + (plus-sign), . (full stop/period/dot), - (hyphen), @ (at-the-rate of symbol), and _ (an underscore). The tags, *Qubole*, and *alias* are reserved for use by Qubole (see [Qubole Cluster EC2 Tags](#)). Tags beginning with *aws-* are reserved for use by Amazon. Click **Add EC2 tags** if you want to add more than one custom tag. A maximum of five custom tags can be added per cluster.

EC2 Settings provides more information. See [Manage Roles](#) for information on how permissions to access features are granted and restricted.

Modifying Hadoop Cluster Settings In **Hadoop Cluster Settings**, you can:

- Specify the **Default Fair Scheduler Pool** if the pool is not submitted during job submission.
- Use **Recommended Configuration** for the selected slave instance type unless it is overridden.
- Set the new **Fair Scheduler Configuration** values to override the default values.
- Override the Hadoop configuration by entering the variables in the **Override Hadoop Configuration Variables** text field.

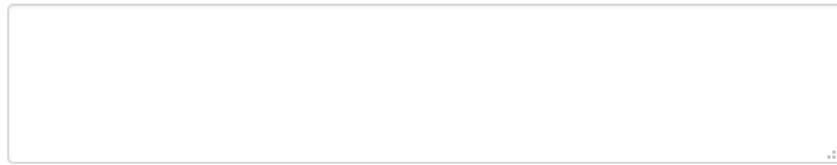
Hadoop-specific Options provides more description about the options.

Note: Recommissioning can be enabled on Hadoop-1 and Hadoop-2 clusters as an Override Hadoop Configuration Variable. See [Enable Recommissioning on Hadoop-1 and Hadoop-2 Clusters](#) for more information.

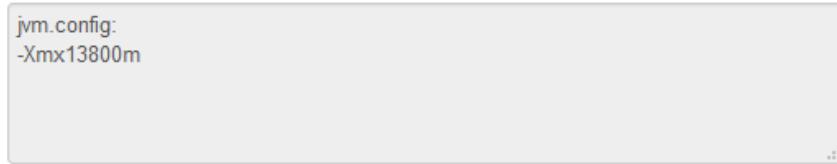
If cluster type is Presto, you must set Presto Settings as shown in the following figure.

Presto Settings See Docs

Override Presto Configuration



Recommended Configuration



See [Presto Cluster Configuration](#).

If the cluster type is Spark, spark configuration is set in the Recommended Spark Configuration, which is in addition to Hadoop Cluster Settings.

If the cluster type is HBase, you must set HBase cluster settings. See [Configuring HBase Clusters](#) for more information.

Modifying Security Settings Qubole Public Key cannot be changed. Setting Customer Public SSH Key is optional.

Persistent security groups - This option overrides the account-level security group settings. By default, this option is not set but it inherits the account-level persistent security group, if any. Use this option if you want to give additional access permissions to cluster nodes.

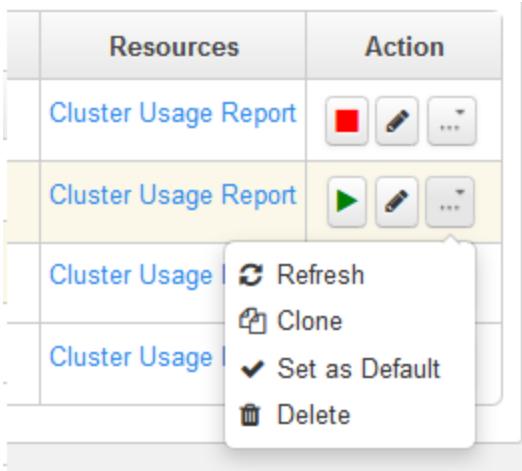
Select **Enable encryption** to encrypt ephemeral storage. It is an option to encrypt the data at rest on the node's ephemeral (local) storage. This includes HDFS and any intermediate output generated by Hadoop. The block device encryption is setup before the node joins the cluster and can increase the bring up time of the cluster.

After modifying the various options of a cluster, click **Save**. Click **Cancel** to restore the previous settings.

Setting a Cluster as Default When a cluster is set as default, any hadoop/hive/pig query/job without an explicitly

specified label uses this cluster. To set it as a default cluster, click the ellipse icon  listed in the **Action** column.

You get a list of options as shown in the following figure.



Select **Set as Default** to set a cluster as default.

Performing Cluster Operations *Cluster Operations* describes about performing operations such as how to start/terminate, create, modify, delete, and clone a cluster.

Managing Groups

In the **Control Panel** page, click **Manage Groups** to create and modify user groups. The **Manage Groups** page is displayed as shown in the following figure.

Manage Groups			
Group Name	Number of Users	Roles associated with this group	Actions
system-admin	1	system-admin	
system-user	0	system-user	

Click the downward arrow in the **Action** column to modify or see users in a group. Click **Modify** and the **Manage Group Members and Group Roles** page is displayed as shown in the following figure.

Manage Groups > Manage Group Members and Group Roles

Manage Group Members and Group Roles	
Group Name	system-admin
Select Group Members	Qubole J (qubolej@qubole.com)
Select Group Roles	system-admin
Update Cancel	

Add/remove users in the **Select Group Members** field and add/remove roles in the **Select Group Roles** field. Click **Update** after making the changes.



Add a group by clicking the add icon . The **Create a New Group** page is displayed as shown in the following figure.

Manage Groups > Create a New Group

The screenshot shows a user interface for creating a new group. At the top left is a small icon of a group of people. To its right, the text "Create a New Group" is displayed. Below this, there are three input fields: "Group Name" containing "Name of the new group", "Select Group Members" containing "Start typing members' names", and "Select Group Roles" containing "Start typing role names". At the bottom of the form are two buttons: a blue "Create Group" button and a grey "Cancel" button.

Enter a name in the **Group Name** text field. Add users from the **Select Group Members** and roles in the **Select Group Roles** fields. Click **Create Group**. Click **Cancel** to not save the group.

Managing Hive Bootstrap

Hive Bootstrap is useful when every hive query that is run inside an account needs to perform a series of steps such as:

- Adding jars
- Defining temporary functions
- Setting Hive parameters
- Mapreduce settings for Hive queries

The Hive bootstrap is run before each query is submitted to QDS.

For example, to use test.py in all sessions, define a bootstrap as shown below.

```
add file s3n://prod.qubole.com/ec2-user_hu_6/scripts/test.py;
```

Hive bootstrap can be defined in two ways by:

- **Base Bootstrap Location:** It is an S3 location that contains the bootstrap file. See [Adding a Bootstrap File in a Base Bootstrap Location](#) for more information.
- **Bootstrap override editor:** It can be used to manually write/edit entries. See [Adding Settings in the Bootstrap Override Editor](#) for more information.

Base Bootstrap is useful when you want to use the same bootstrap configuration in multiple accounts. If the bootstrap file in S3 is updated, it affects all accounts that use this file. Settings in the bootstrap override editor override the settings in the base bootstrap location. Hence, account-specific settings are added in the bootstrap override editor.

See [Hive Bootstrap](#) for more information.

To set a hive bootstrap configuration, click **Hive BootStrap** in **Control Panel**.

Clicking **Hive Bootstrap** displays **Bootstrap Settings** that provides a text field to specify the absolute path of an S3 location that contains the bootstrap file. It also provides an editor where you override the configuration that must be executed before a command/query gets executed for an account.

Add the bootstrap configuration file into an S3 location if it is not added. Enter that S3 location of the bootstrap file in the **Base Bootstrap Location** text field and click **Save**. The absolute path is displayed as shown in the following figure.

The screenshot shows the 'Bootstrap Settings' page. At the top, there is a 'Base Bootstrap Location' input field containing 'mybucket/hive/scripts/hive_bootstrap_base_script'. Below it is a 'Save' button. Underneath the input field is a 'Bootstrap Override Script Editor' section with a code editor containing the following script:

```
set mongo.input.split.create_input_splits=false;
set mongo.mapper.count=5;
```

At the bottom left of the editor is an 'Edit' button.

Click the **model** button icon  that is next to the **Base Bootstrap Location** text box to see the contents of a bootstrap file.

By default, **BootStrap Override Script Editor** is blank and displays a configuration only when a user sets a bootstrap configuration for the current account.

Note: To change the boot configuration for multiple accounts, modify the boot configuration file with the new configuration at the S3 location (base bootstrap location). The new hive bootstrap configuration gets applied to all accounts.

To set a different bootstrap configuration for the current account, click **Edit** in **Bootstrap Override Script Editor**. Update a different bootstrap configuration and click **Save**. The following figure shows an example of overriding a bootstrap configuration.

The screenshot shows the 'Bootstrap Settings' page again. The 'Base Bootstrap Location' field remains the same. The 'Bootstrap Override Script Editor' now contains a different script:

```
set mongo.input.split.create_input_splits=true;
set mongo.mapper.count=5;
```

At the bottom of the page are three buttons: 'Save', 'Clear', and 'Cancel'.

Click **Clear** to reset. Click **Cancel** to retain the previously set bootstrap configuration.

Whitelisting IP Addresses

Whitelisting IP addresses allows users of an account to login only from certain IP addresses.

Note: Write an email to solutions@qubole.com to enable this feature for the account.

You can add the IP addresses to whitelist them in the [Control Panel](#).

To whitelist an IP address, perform the following steps:

1. On the QDS user interface, navigate to the [Control Panel](#). Click the **Whitelist IP** tab. If there is no whitelist IP address added, it is as shown in the following figure.

Whitelisted IP		
ID	IP CIDR	ACTION
No data is available.		

2. Click the add icon  to add an IP address to the whitelist.

A dialog with **OK** and **Cancel** buttons are displayed. Click **OK** to add a new IP address. The dialog to add a new IP is displayed. Enter the IP address in the IP address in the IP CIDR text field as illustrated in the following figure.

Add a new IP to whitelist

IP CIDR	11.111.111.111
Cancel	Add Entry

Caution: Once you add IP addresses to whitelist, QDS allows account access only when it is accessed with the whitelisted IP addresses.

Click **Add Entry** to save the IP address to the whitelist. After clicking **Add Entry**, the **Whitelist IP** tab displays the newly-added IP address as shown in the following figure.

IP whitelisted successfully.

Whitelisted IP		
ID	IP CIDR	ACTION
1	11.111.111.111	

Click the refresh icon  to refresh the whitelist. Repeat step 2 to add another IP address to whitelist.

In the **Action** column, click **Delete** to remove an IP address from the whitelist. A dialog with **OK** and **Cancel** buttons are displayed. Click **OK** to delete a whitelisted IP address. After you click the **OK** button, the whitelisted IP address is not seen in the **Whitelist IP** tab as shown in the following figure.



Managing My Accounts

Qubole lets you have multiple accounts and only one default account. Click **My Accounts** to add and manage accounts. You are logged into the account that is marked as default here.

My Accounts is displayed as shown in the following figure.

My Accounts						Override AWS Credentials	+
Account Name	Account Id	Status	My Groups	API Token	Action		
quboledemo	234	✓	explore, qubole-grp	Show			
dev-perftest 	1	✓	system-admin	Show Reset			

Switch to a different account if you prefer to use it. You can also click the accounts drop-down list on the top right corner of the QDS UI, to switch to a different account and clone an account.

Only the API token of the default account can be reset. API tokens are used to authenticate with the API. An API token is for a user per account. This implies that a user, who is part of 10 accounts has 10 API tokens. A user with a single account has one API token.

An API token can be used to schedule jobs using external schedulers such as cron but it is not required when jobs are scheduled using the Qubole scheduler. The jobs are shown by the user whose API is being used. If it is required to use a single user for all scheduled jobs, create a common user to run them.

In the **Action** column, click the downward arrow to select **Override default AWS settings** and **Set this account as Default** (only visible for a non-default account) as required.

Click **Override AWS Credentials** to override the AWS settings for multiple accounts.

Override AWS settings for multiple accounts is displayed as shown in the following figure.

Override AWS settings for multiple accounts

AWS Access Key	<input type="text" value="Enter your AWS access key"/>						
AWS Secret Key	<input type="text" value="Enter your AWS secret key"/>						
<table border="1"><thead><tr><th></th><th>Account Name ↴</th></tr></thead><tbody><tr><td><input type="checkbox"/></td><td>quboledemo</td></tr><tr><td><input type="checkbox"/></td><td>dev-perftest</td></tr></tbody></table>			Account Name ↴	<input type="checkbox"/>	quboledemo	<input type="checkbox"/>	dev-perftest
	Account Name ↴						
<input type="checkbox"/>	quboledemo						
<input type="checkbox"/>	dev-perftest						
<input type="button" value="Cancel"/>	<input type="button" value="Save Settings"/>						

Enter the new keys in the **AWS Access Key** and **AWS Secret Key** text fields for the account that you want to change. Click **Save Settings** after changing the keys. Click **Cancel** to go back to the **My Accounts** page.

Click the add icon  to create a new account. The **New Account** page is displayed as shown in the following figure.

Manage Accounts > New Account

	Create a New Account
--	----------------------

Account Info

New account name	<input type="text" value="Name of New Account"/>
------------------	--

Storage credentials

Default S3 location	<input type="text" value="eg. mybucket/qubole/account"/>
AWS Access key	<input type="text" value="eg. AW23GS0LOPPG3JW2ZX9"/>
AWS Secret key	<input type="text" value="eg: f9hBE0pGkuv5imKnY0t9O3"/>

Compute credentials

<input type="checkbox"/> Same as storage credentials	
AWS access key	<input type="text" value="eg. AW23GS0LOPPG3JW2ZX9"/>
AWS secret key	<input type="text" value="eg: f9hBE0pGkuv5imKnY0t9O3"/>
AWS region	<input type="button" value="us-east-1"/>

By Clicking Create Account, you agree to our [Terms and Conditions](#).

<input type="button" value="Create Account"/>	<input type="button" value="Cancel"/>
---	---------------------------------------

To create a new account, perform the following steps:

- In **Account Info**, enter a name in the **New Account Name** text field.
- In **Storage credentials**, enter the corresponding values in **Default S3 location**, **AWS Access Key**, and **AWS Secret Key**
- In **Compute credentials**, select **Same as storage credentials** if you prefer to use the same credentials. Else, enter the corresponding values in **AWS access key** and **AWS secret key**. Select the AWS region from the drop-down list. Read **Terms and Conditions** to be aware of them.

Click **Create Account** to create a new account. *Clicking Create Account implies that you agree to the Terms and Conditions.* Click **Cancel** to not create the account.

Note: Use the tooltip to know more information on each field or checkbox.

Managing Profile

To edit your profile and change the password, click **My Profile** in the **Control Panel** page. The **My Profile** page is displayed as shown in the following figure.

The screenshot shows a web-based form titled "My Profile". It contains two text fields: "Email ID" with the value "qubolej@qubole.com" and "Full Name" with the value "Qubole J". Below these fields are two buttons: "Change Password" and a blue "Save" button.

Modify the name in the **Full Name** text field. To change the password, click **Change Password**. Else, click **Save**. When you click **Change Password**, you get additional text fields to change the password as shown in the following figure.

The screenshot shows the same "My Profile" page, but now with four additional text fields for password management: "Current Password", "New Password", and "Confirm Password", all currently empty. The "Save" button remains at the bottom.

Fill the **Current Password** and **New Password** text fields. Reconfirm the new password in the **Confirm Password** text field. Click **Save** to change the full name and password.

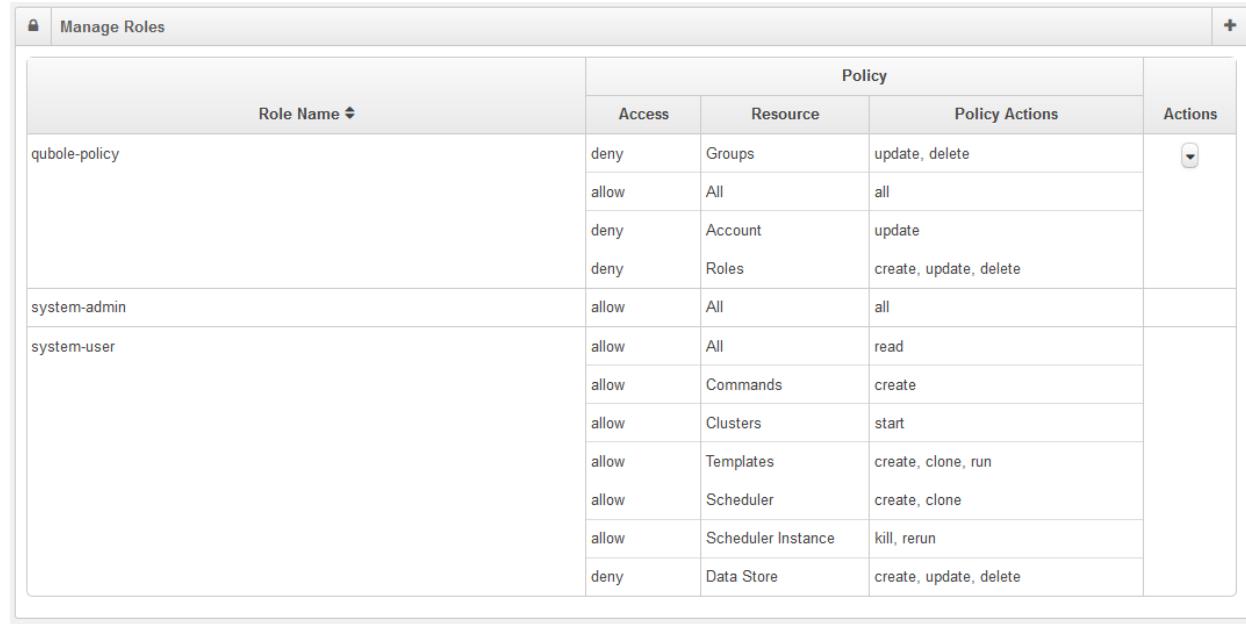
Managing Roles

Roles work in the following way:

- System-admin - It is a system-defined role, which has full access to all resources. People, who are part of the group of the same name get full access to all resources in a Qubole account.
- System-user - It is also a system defined role, whose access is restricted. It allows people, who are part of the group with this role to read all Qubole resources (such as commands and templates), create commands in **Analyze**, create, clone and run Templates, create, clone Schedules, kill and rerun Scheduler Instances. Create, update, and delete operations on data stores in **Explore** are denied. Create, clone, update, and delete cluster operations are also denied.

Apart from these system defined roles, you can create roles with a definition allowing granular control over the features of the Qubole user interface. When a single resource with allow and deny restrictions across two roles in the same group is applied, the least restrictive policy applies. For example, if Role1 denies read access to commands and Role2 allows read access to commands. If Role1 and Role2 are part of a group named Group1, then, while applying these rules, Role2's policy gets precedence over Role1's policy in Group1.

To manage roles by granting/denying access to the resources by setting policy actions, click **Manage Roles** in the **Control Panel** page. **Manage Roles** is displayed as shown in the following figure.

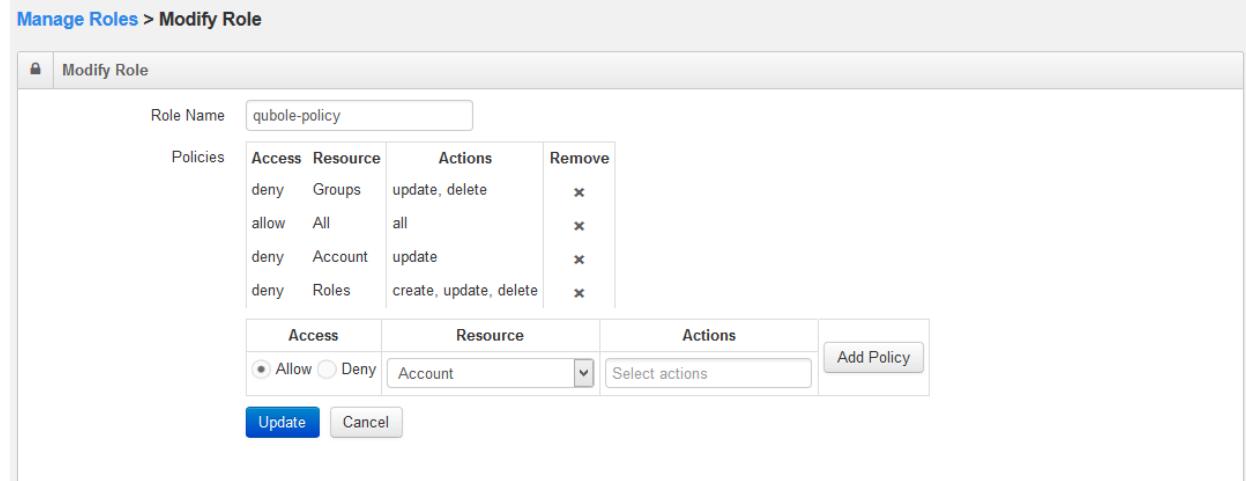


The screenshot shows a table titled "Manage Roles". The columns are "Role Name", "Policy", and "Actions". The "Policy" column has sub-columns "Access", "Resource", and "Policy Actions". The "Actions" column contains a downward arrow icon.

Role Name	Policy			Actions
	Access	Resource	Policy Actions	
qubole-policy	deny	Groups	update, delete	
	allow	All	all	
	deny	Account	update	
	deny	Roles	create, update, delete	
system-admin	allow	All	all	
system-user	allow	All	read	
	allow	Commands	create	
	allow	Clusters	start	
	allow	Templates	create, clone, run	
	allow	Scheduler	create, clone	
	allow	Scheduler Instance	kill, rerun	
deny	Data Store	create, update, delete		

Modifying a Role

In the **Action** column, click the downward arrow and select **Modify** to edit a role.



The screenshot shows a "Modify Role" dialog box. It has a "Role Name" field set to "qubole-policy". Below it is a table titled "Policies" with columns "Access", "Resource", "Actions", and "Remove". There are four rows in the table. At the bottom, there is a section with "Access" (radio buttons for "Allow" and "Deny"), "Resource" (dropdown menu set to "Account"), "Actions" (button "Select actions"), and a "Add Policy" button. Below this are "Update" and "Cancel" buttons.

Policies			
Access	Resource	Actions	Remove
deny	Groups	update, delete	
allow	All	all	
deny	Account	update	
deny	Roles	create, update, delete	

Do the changes and click **Update**. Click **Cancel** to retain the previous setting.

Creating a Role



Click the add icon to create a new role. The **Create a New Role** page is displayed as shown in the following figure.

Manage Roles > Create a New Role

Role Name	<input type="text" value="Name of the new role"/>		
Policies	Access	Resource	Actions
	<input checked="" type="radio"/> Allow <input type="radio"/> Deny	<input type="text" value="Account"/>	<input type="button" value="Select actions"/> <input type="button" value="Add Policy"/>
	<input type="button" value="Create Role"/> <input type="button" value="Cancel"/>		

Enter a name in the **Role Name** text field. Add a new policy to the role. In the **Policies** field, select either **Allow** or **Deny** access.

Select a resource from the **Resource** drop-down list. Select a action, **all/read/update** from the **Action** drop-down list.

Qubole now supports create, update, delete, start, terminate, and clone actions that a role can perform on a cluster as shown in the following figure.

Create a New Role

Role Name	<input type="text" value="Name of the new role"/>		
Policies	Access	Resource	Actions
	<input checked="" type="radio"/> Allow <input type="radio"/> Deny	<input type="text" value="Clusters"/>	<input type="button" value="Select actions"/> <input type="button" value="Add Policy"/>
	<input type="button" value="Create Role"/> <input type="button" value="Cancel"/>	<input type="button" value="all"/> <input type="button" value="create"/> <input type="button" value="read"/> <input type="button" value="update"/> <input type="button" value="delete"/> <input type="button" value="start"/> <input type="button" value="terminate"/> <input type="button" value="clone"/>	

Click **Add Policy** to add a policy for the new role. The following figure shows an example of adding a policy to a role.

The screenshot shows the 'Create a New Role' interface. At the top, there's a lock icon and the title 'Create a New Role'. Below that, a 'Role Name' field contains 'ReadUser'. Under 'Policies', there's a table with four columns: Access, Resource, Actions, and Remove. One row shows 'allow' for 'Clusters' with 'read' in the Actions column. Below this is a more detailed policy editor with tabs for Access, Resource, and Actions, showing 'Allow' selected. A 'Select actions' dropdown and an 'Add Policy' button are also present. At the bottom are 'Create Role' and 'Cancel' buttons.

Click **Create Role** after adding the policy and name. Click **Cancel** to not create the role.

Managing Sessions

A session is a group of interactions that occur on a website within a given time frame.

Hive allows you to embed the code (python scripts, shell scripts, Java functions) into SQL queries. This is useful as a way to quickly enhance the language to support functionality that is not natively present in HiveQL. See [Log Parsing](#) and [Language Manual](#) for examples of SQL queries using this functionality.

Qubole mimics this functionality by associating every command in QDS with a session. Sessions let you create temporary data sets (that last for the duration of the session) as well as configure specific session parameters, which can be used to tune query behavior. Sessions also let you add code as scripts to Qubole so that they can be used to run transformations within HiveQL. Through sessions, you can isolate the effects of these parameters, temporary datasets and user defined transformations across various workloads. Sessions have lifetimes that can be edited (default value 2).

To ensure that the scripts are accessible to Qubole's Hive clusters, upload them to Amazon's S3 storage service. The script can be in any S3 location that is readable using the keys associated with account. However, Qubole recommends that you place scripts within a default location's scripts folder.

For example, if the default location is `prod.qubole.com/ec2-user_hu_6`, Qubole suggests placing scripts in `s3://prod.qubole.com/ec2-user_hu_6/scripts/`. Once a script has been uploaded to a S3 location, add it to your session by running a **add file ...** command. This command automatically gets added to the session, if it is successful and this script is usable in other queries that are part of this session. Consider a simple `test.py` script, which has the following code:

test.py

```
#!/usr/bin/python

import sys
line = sys.stdin.readline();
while line:
    print line
    line = sys.stdin.readline();
sys.exit(0)
```

If the default location is `prod.qubole.com/ec2-user_hu_6`, upload `test.py` to `prod.qubole.com/ec2-user_hu_6/scripts/test.py`. Once this has been uploaded you can issue:

```
add file s3n://prod.qubole.com/ec2-user_hu_6/scripts/test.py;
//and then use this in a query as
```

```
select count (*)
from (select transform(a) using 'python test.py' from tb) V
```

To create and manage sessions, click [Sessions](#) in the **Control Panel** page. Create a session by clicking the add icon



at the top right corner of the **Sessions** page.

A dialog to create a session appears as shown in the following figure.



Select a cluster on which you want to create a session from the drop-down list and click **Create Session**. A session is created and a sample session is shown in the following figure.

Sessions					
Id	Cluster Id	Start Time	Duration	Commands	Action
300259	4633	10:57 AM March 05	2 hours	1	

The **Sessions** page contains five columns as listed below:

- **Id:** Denotes the session ID.
- **Cluster Id:** Denotes the Cluster ID on which session is created.
- **Start Time:** Denotes the start time of the session
- **Duration:** The default duration time is 2 hours. You can modify it by setting it in hours. The supported value range is 1 to 6 hours.
- **Commands:** The number of commands that was run previously is shown in this column. To see the commands, click the number. It takes you to [Compose](#) of the [Analyze](#) page.
- **Action:** Click the downward arrow in the **Action** column. You get a list of actions as shown in the following figure.

Sessions					
Id	Cluster Id	Start Time	Duration	Commands	Action
300259	4633	10:57 AM March 05	2 hours	1	

[View Commands](#)
[Change Duration](#)
[Deactivate](#)
[Delete](#)

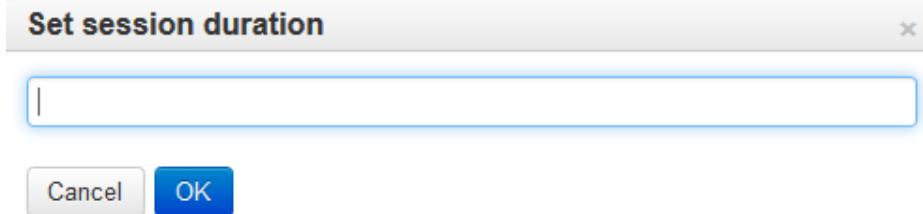
You can perform the following operations on an existing session:

- **View Commands:** Select this action to see the commands that are running in the session. Once you select this action, the **Session Details** dialog appears. Click the downward arrow **Action** column as shown in the following figure.



You can prefer to go to the **Analyze** page or **Remove** the command.

- **Change Duration:** Select this action to change the session duration. The **Set Duration** dialog appears as shown in the following figure.

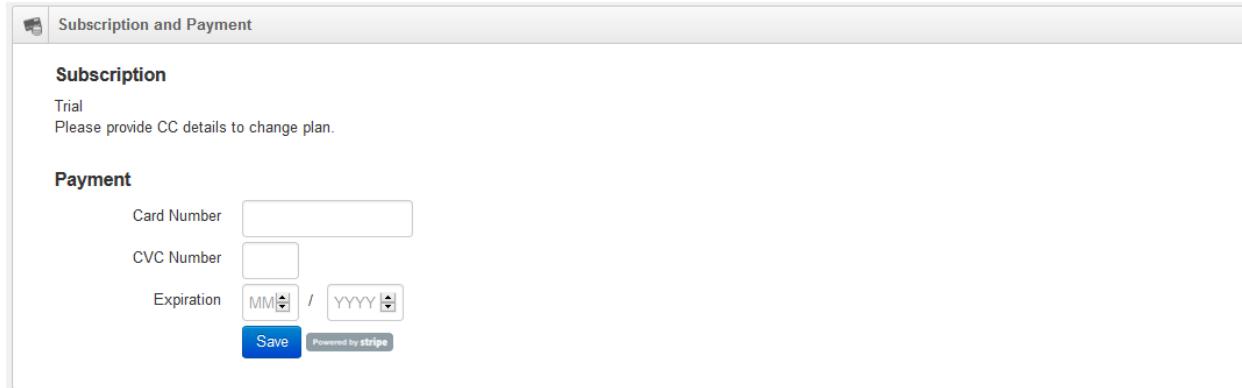


The default is 2 hours and the value range is 1-6 hours. Set the value and click **OK**. Click **Cancel** to restore the previous/default settings.

- **Deactivate:** Select this action to deactivate a session. You can activate a deactivated session within two hours. To activate a deactivated session, click downward arrow in the **Action** column and click **Activate**.
- **Delete:** Select this option to delete the session.

Managing Subscription and Payment

To change the subscription plan, click **Subscription and Payment** in the **Control Panel** page. **Subscription and Payment** is displayed as shown in the following figure.



Subscription shows you the type of subscription.

In **Payment**, enter the **Card Number**, **CVC Number**, and **Expiry Date** of your debit/credit card and click **Save**. Click **Cancel** to not subscribe.

If you want to change the plan, select a plan from the **Choose your plan** drop-down list. The different plans that are available are as shown in the following figure.

Your 15 day trial has expired

Thank you for using Qubole Data Service (QDS). Our records show that your trial period has expired.

For further usage please choose your plan.

Choose your plan

Elastic, (\$0.11/QCUH, 3 Free Users, \$0.22/Invocation)	<input type="button" value="▼"/>
Elastic, (\$0.11/QCUH, 3 Free Users, \$0.22/Invocation)	
Enterprise, \$5900/Month (28 User Licenses, 40000 QCUH, 2000 Invocations)	
Basic, \$1050/Month (8 User Licenses, 5000 QCUH, 750 Invocations)	

Questions? Contact our sales at 1-855-423-6674 or email: sales@qubole.com

Click **Update** after selecting the plan.

Managing Users

To manage users, click **Manage Users** in the **Control Panel** page. The **Manage Users** tab is displayed as shown in the following figure.

The screenshot shows a user interface for managing users. At the top, there are two tabs: "Manage Users" (selected) and "Pending Users". Below the tabs is a section titled "Account: qubolej" with a sub-instruction: "New user can join your account while signing up using this link: https://api.qubole.com/users/sign_up?join_account=4J96MWNAKG9MKCKE3064". A table below lists one user:

User Full Name	Email Address	Groups	Last Login	Action
Qubole J	qubolej@qubole.com	system-admin	2015-03-06 10:59	

A new user can register by using the link, **Signing Up as a New User**. The link is also shown in the figure provided above that is, the **Manage Users** page. Click **Action** to modify or delete the user. Click the user icon  to invite a new user.

The following page to invite a new user is displayed as shown in the following figure.

The screenshot shows a "Manage User > Invite User" page. It has a header with a "Manage User" icon and the title "Manage User > Invite User". Below the header is a navigation bar with "Manage User" and "Invite User" tabs, where "Invite User" is selected. The main form contains two fields: "User Email" (text input field) and "Groups" (text input field with placeholder "Start typing group names"). At the bottom are two buttons: "Send" (blue) and "Cancel".

Enter the email address in the **User Email** text field. Select a group from the available list in **Groups**. Click **Send** to invite a user. Click **Cancel** to reenter the email and group information/not invite any user.

2.2.6 Notebook

Introduction to Notebooks

Qubole has now introduced Notebooks to write commands to run on HBase, Presto and Spark clusters.

Accessing Notebooks

On the QDS user interface (UI), navigate to the **Notebook** feature and the notebook homepage is displayed. The following figure shows an example of the notebook homepage.

The screenshot shows the QDS Notebook homepage. The left sidebar lists a large number of notebooks, each with a small colored circle indicating its status: green for running, red for down, and grey for not assigned. The notebooks listed are: test_note-Clone, test_note-Clone3, test1, Note, test12, test1-Clone, test2, test2 - Clone, test_last, test_lastClone, test_lastClone1, test_last - Clone2, and test1234. The right panel has a welcome message: "Welcome to Notebook" and "Notebooks provide an interactive interface for data exploration. You can view, run and visualize the results of SQL, Scala, Python, and more in a single, collaborative environment." It also includes a button to "Get started by selecting a notebook from your list."

The left panel displays the list of notebooks. By default, ExampleNote, PrestoExampleNote are available in the list. You can hide/unhide the left panel to see/hide the list of notebooks. A notebook that has a green circle against it indicates that its assigned cluster is running. A red circle against a notebook indicates that its assigned cluster is down. A grey circle against a notebook indicates that it is not assigned to any cluster.

The following topics explain how to manage and use notebooks:

- *Managing Notebooks*
- *Using a Notebook*

Managing Notebooks

You can add, edit, or, delete a notebook and the following topics cover how to manage notebooks:

- *Viewing a Notebook Information*
- *Locking a Notebook*
- *Adding a Notebook*
- *Modifying a Notebook*
- *Cloning a Notebook*
- *Filtering a Specific Notebook from a List*
- *Viewing the Permalink of a Notebook*
- *Deleting a Noteboook*

Viewing a Notebook Information

Click the notebook in the left panel to view its details. The notebook's ID, its type, associated cluster are displayed. The following figure shows an illustration of a notebook's details displayed.

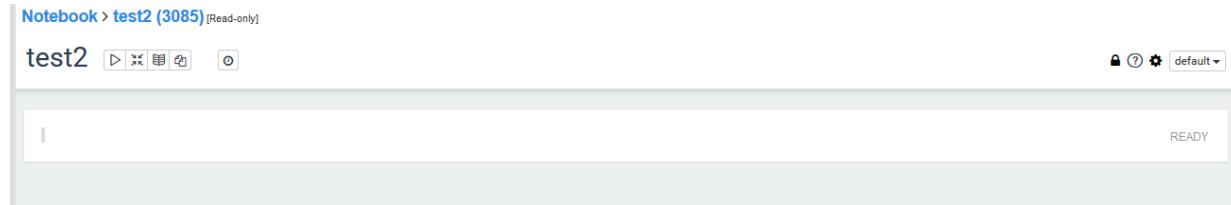
The figure consists of two panels. The left panel shows a list of notebooks with their names and status icons (green circle for running, red circle for down). The right panel shows the same list, but the notebook 'test1234' is selected, which highlights it with a blue background. Below the list, detailed information is provided for the selected notebook: 'Id: 3092' and 'Cluster: 19718 - spark'. Both panels include standard UI elements like a search bar, a '+' button, and a refresh icon at the top, and a toolbar with icons for settings, copy, delete, and refresh at the bottom.

A notebook that contains a green circle against it indicates that its assigned cluster running in the left panel. Click it in the left panel and the note is displayed in the right panel as shown in the following figure.

This screenshot shows the detailed view of the notebook 'test1234'. The left panel is a list of notebooks, and the right panel is a detailed view of 'test1234'. The right panel includes a code editor area with placeholder code, a toolbar with various icons, and status information: 'Interpreters default' and 'Connected' (indicated by a green circle). The notebook 'test1234' is also listed in the left panel.

The notebook shows a green circle against **Connected** status and **Interpreters**.

A notebook that contains a red circle against indicates that its assigned cluster in a down state. Such notebooks are read-only and cannot be used to run a query. A sample of a read-only notebook is as shown in the following figure.



You can edit the name of a read-only notebook. After starting the assigned cluster, this notebook can be used to run queries.

A notebook that contains a grey circle against it indicates that it does not have any assigned cluster. Click it to see more details. A sample of an unassigned notebook is as shown in the following figure.

A screenshot of the Qubole notebook list. On the left, there's a sidebar with "+", "-", and a refresh icon. The main area shows a list of notebooks: "test_note-Clone" (Id: 3076, Cluster: N/A), "test_note-Clone3" (Id: 3077, Cluster: N/A), "test1" (highlighted with a red dot and has a grey circle icon), and "Note". The status bar at the bottom right says "READY".

Locking a Notebook

With Qubole First Class Notebooks, you now have the ability to lock and unlock notes. When you lock a note, you can prevent edits on the note from other users in the account. Once the notebook is ready to be used, you can unlock the note to make it available to all users in the account. This simplifies the user experience in a multiuser Qubole account.

To lock a notebook, click the lock icon, .

When you lock a note, the icon turns unlocked that indicates that note is locked as illustrated in the following figure.

A screenshot of a Qubole notebook interface. The title bar says "ExampleNote". In the top right, there's a blue button with the text "Lock acquired" and a padlock icon. The main area shows a code cell with some text and a warning message about interpreters. The status bar at the bottom right says "FINISHED".

By locking a note, you get exclusive control over it. Other users in the account can only view the note and cannot edit or delete the note until you unlock it. For unlocking a note, click the unlocked icon in the locked note as illustrated in the following figure.

The screenshot shows the Qubole Notebook interface. At the top, there's a header bar with the Qubole logo, 'Notebook', 'Interpreter', and a status indicator 'Connected'. Below the header is a toolbar with icons for back, forward, search, and other notebook operations. The main area contains a note titled 'ExampleNote' with the following content:

```

Introduction
Kmql ## This notebook supports multiple language backend
- default: scala with SparkContext available as sc
- Kmql: markdown
- Ksql: hive on spark by default. hiveContext object is also available by default. If you specify zeppelin.spark.useHiveContext as false in interpreter settings then this will use spark sql and sqlContext object becomes available
- Kpyspark: pyspark
- Ksh: shell

Interpreters are loaded lazily and hence takes some time the first time.

```

The status bar at the bottom right shows 'FINISHED'.

Adding a Notebook

As a prerequisite, to add a notebook of a particular cluster type (HBase/Presto/Spark), the QDS account must contain atleast one cluster of that particular cluster type.



To add a notebook, click the add icon that is in the left panel.

The **Add Notebook** dialog is displayed. Add a name of the notebook as shown in the following figure.

The 'Add Notebook' dialog box has the following fields:

Name	spark-notebook
Type	Spark
Cluster	Select 8036 spark

At the bottom are 'Cancel' and 'Save' buttons.

Select the type from the drop-down list that shows **Spark** by default. The other types in the drop-down list are **HBase** and **Presto**. The **Cluster** field displays the available clusters of the selected type. Select a cluster to which you want to assign the notebook. Click **Save** to add the notebook. Click **Cancel** to not add it. A unique notebook ID is assigned to the newly created notebook.



Click the to refresh the list of notebooks or see any changes that are recently done.

Modifying a Notebook



To edit/configure an existing notebook, click the settings icon displayed from the list of icons for a given notebook.

The **Configure Notebook** dialog is displayed as shown in the following figure.

Configure Notebook

Id	3092
Name	test1234
Type	Spark
Cluster	● 19718 spark

Cancel **Save**

The cluster type cannot be changed. Change the name or the assigned cluster by selecting the one from the drop-down list if any.

Cloning a Notebook

You can clone a notebook if you want the same settings. Select a notebook and click the clone icon  from the list of icons displayed.

The **Clone Notebook** dialog is displayed as shown in the following figure.

Clone Notebook

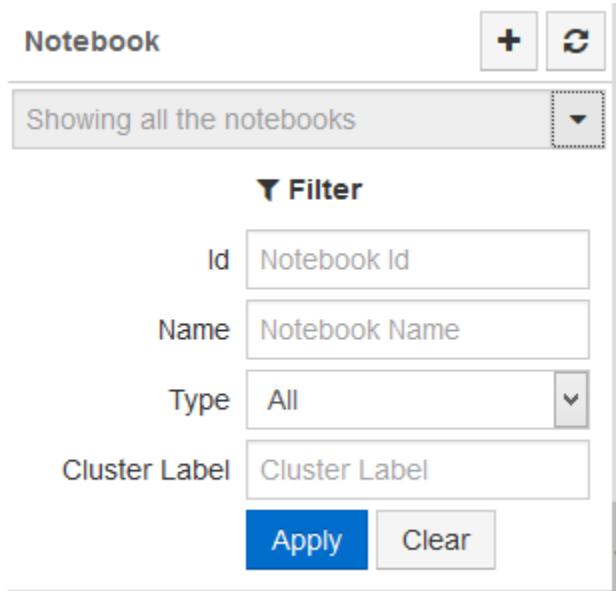
Cloned from Id	3092
Name	test1234 - Clone
Type	Spark
Cluster	● 19718 spark

Cancel **Save**

You can change the name of the notebook and its assigned cluster.

Filtering a Specific Notebook from a List

If you want to see a specific notebook from a list of notebooks, pull the downward arrow in the left pane that is against the search field. The filter is displayed as shown in the following figure.



Type the notebook ID, name, cluster type, or cluster label to filter it from the list and view its details.

Viewing the Permalink of a Notebook

To see the permalink of a notebook, click the **permalink icon**  from the list of icons displayed.

The permalink of the notebook is displayed.

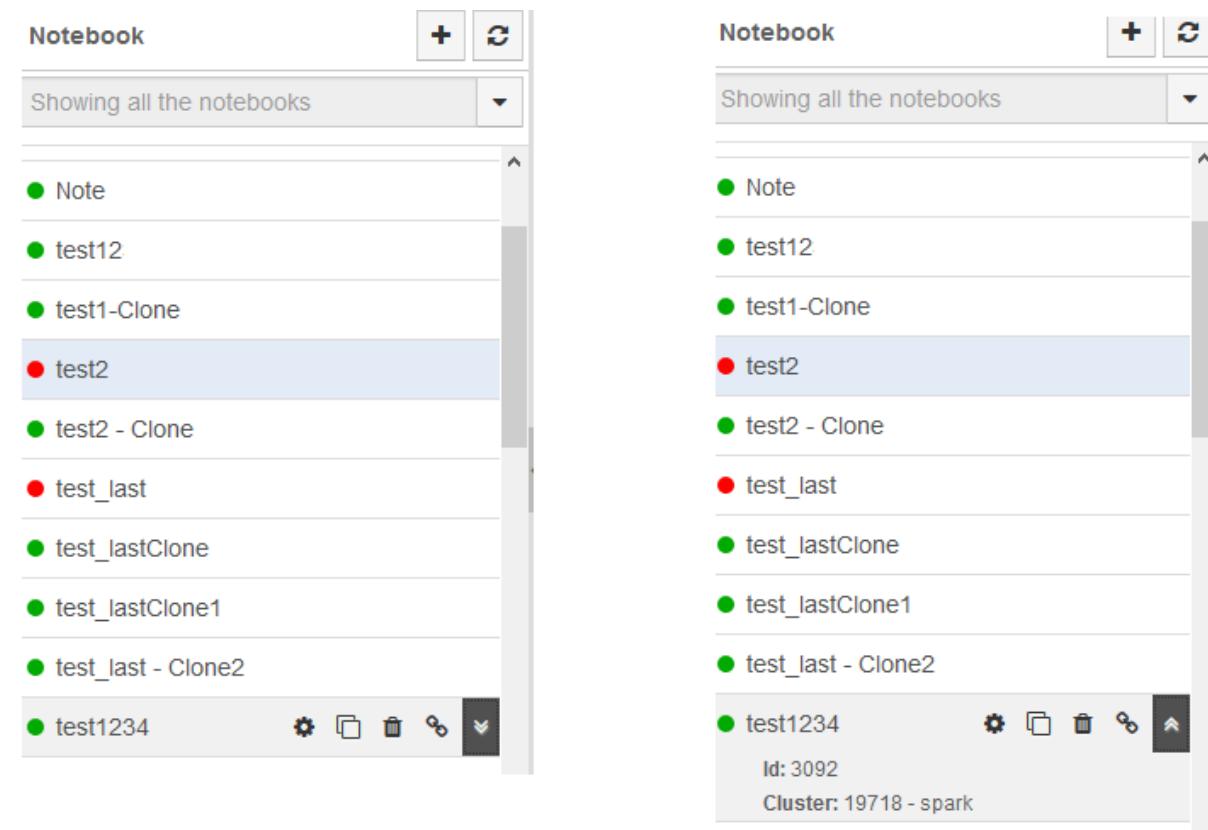
Deleting a Noteboook

To delete a notebook, select a notebook and click the delete icon  from the list of icons displayed.

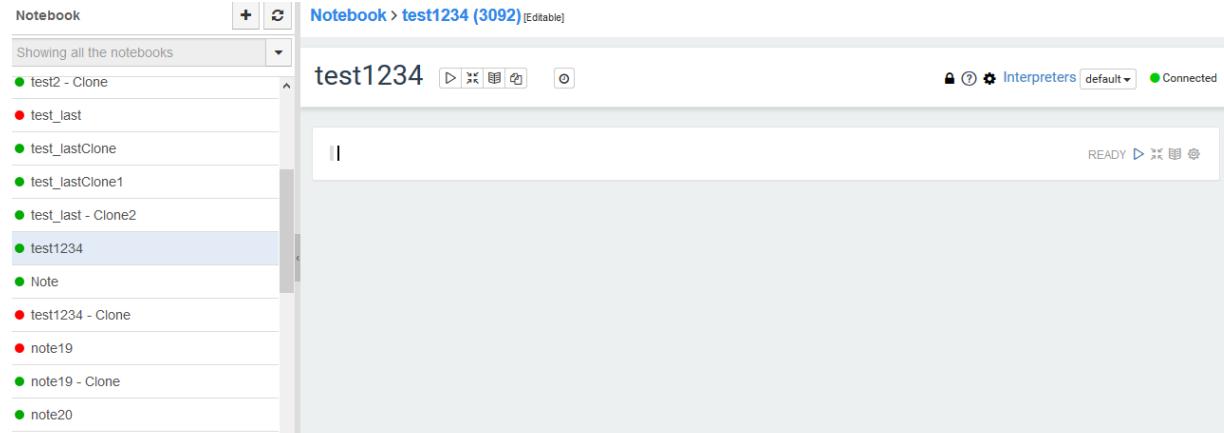
A dialog is displayed to confirm if you want to delete the notebook. Click **OK** to delete it or **Cancel** if you want to retain it.

Using a Notebook

Click the notebook in the left panel to view its details. The notebook's ID, its type, associated cluster are displayed. The following figure shows an illustration of a notebook's details displayed.



Only a notebook with its cluster running can be used to run queries. A sample of a notebook with its cluster up is as shown in the following figure.



Using an HBase Notebook

Select an HBase notebook from the list of notebooks and ensure that its assigned cluster is up to use it for running queries. See [Using HBase Shell](#) for more information.

Using a Spark Notebook

Select a Spark notebook from the list of notebooks and ensure that its assigned cluster is up to use it for running queries. See [Running Spark Applications in Notebooks](#) for more information.

Using a Presto Notebook

Select a Presto notebook from the list of notebooks and ensure that its assigned cluster is up to use it for running queries. The process to run query is the same that is followed to run a Spark or HBase query.

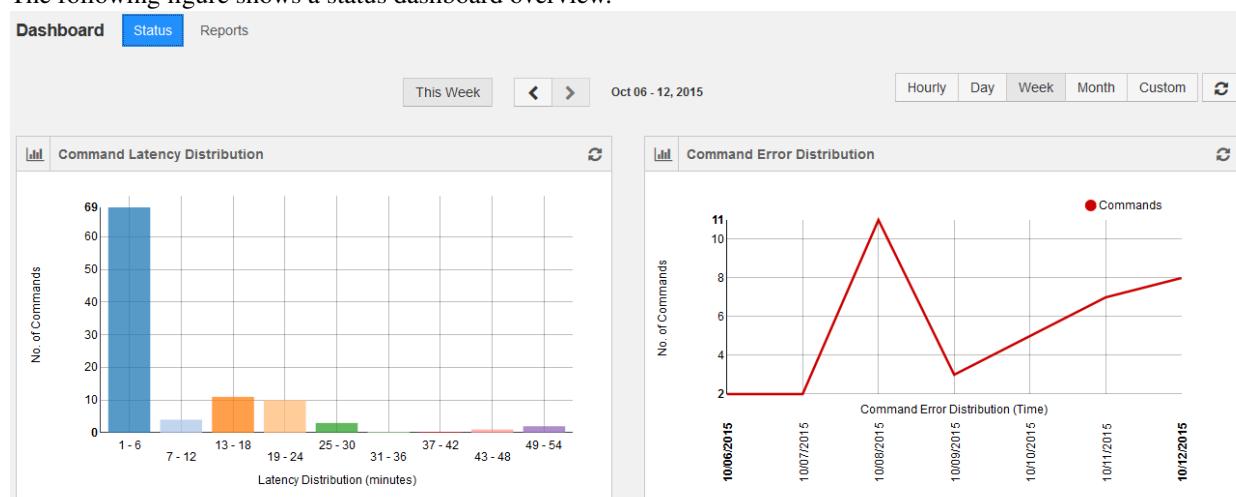
2.2.7 QDS Usage Dashboard Overview

The QDS [Usage Dashboard](#) provides a graphical representation of an account-specific status of commands run and cluster nodes by default in the **Status** tab by default. The **Reports** tab can be used to generate reports such as all commands, cluster metrics, and cluster usage. The QDS **Status** and **Reports** tab are explained in the following topics:

- [QDS Usage Status Dashboard](#)
- [Generating Reports using QDS Dashboard](#)

QDS Usage Status Dashboard

The following figure shows a status dashboard overview.



By default it provides weekly status and the status can be viewed by selecting the frequency: **Hourly**, **Day**, **Month**, and **Custom**. Selecting **Custom** provides an option to select a date range.

The dashboard provides the following status:

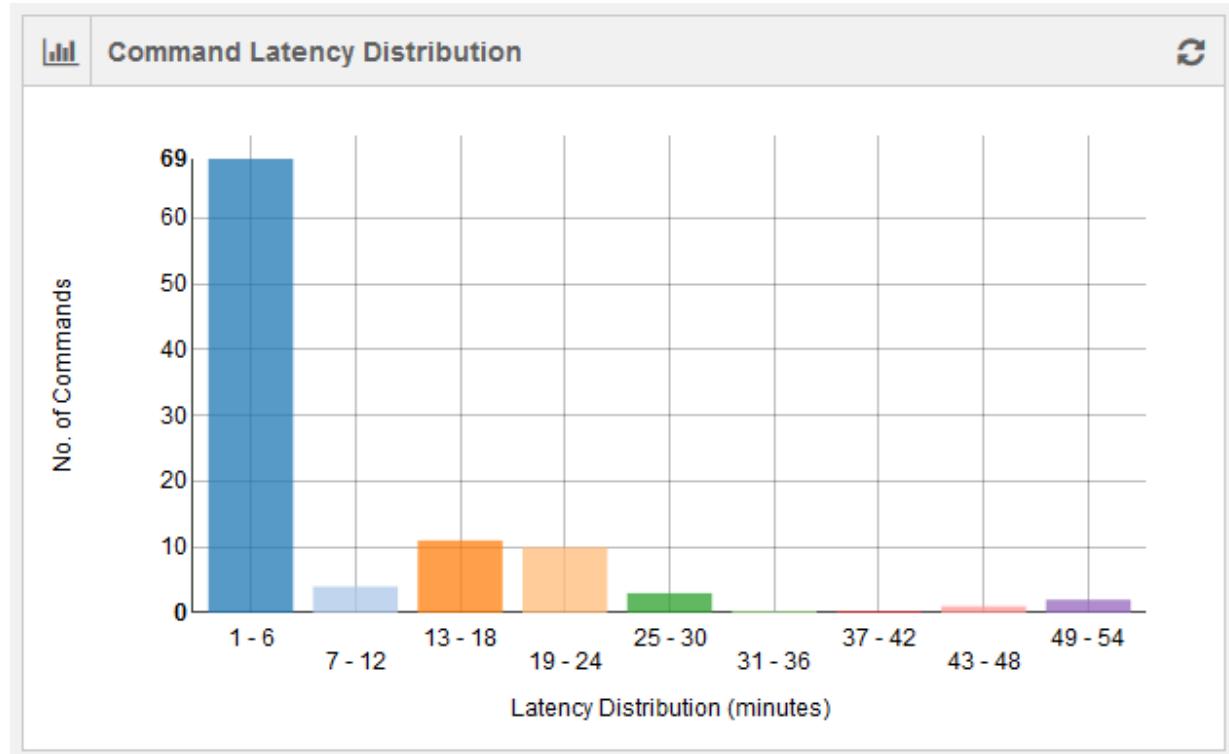
- [Command Latency Distribution Status](#)
- [Command Error Distribution Status](#)
- [Cluster Nodes vs Time Status](#)
- [Average Command Latency Status](#)
- [Cluster Metrics QCUH Status](#)
- [Command Status](#)

- *Job Instance Status*
- *Longest Command/Job Instances Status*
- *Top Most Failing Active Jobs Status*
- *Leaderboard Status*

Command Latency Distribution Status

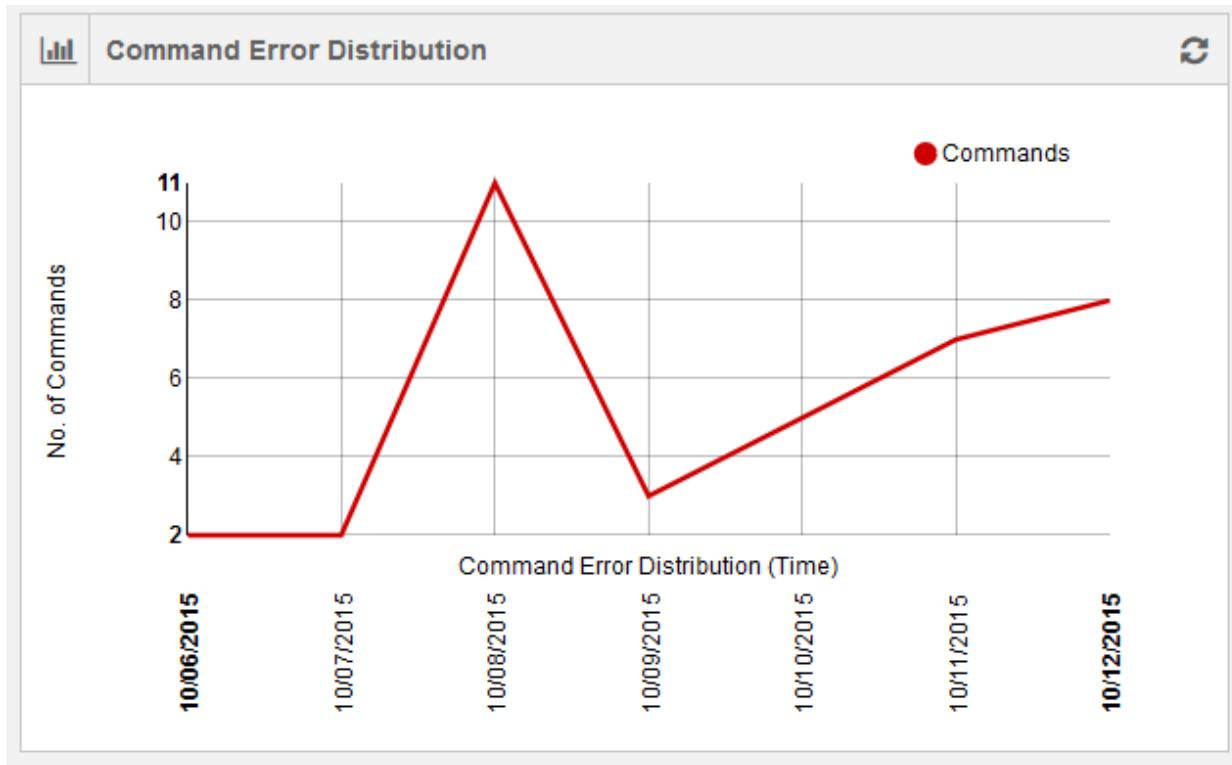
The command latency distribution for an account is shown as a bar graph. The number of latent commands is plotted against time (in minutes). The command duration for a set of commands run at a given time is considered. The difference between maximum and minimum command execution duration is calculated. The latency denominator is calculated by dividing the difference in the minimum and maximum command duration by 10. This graph contains 10 data points.

The following figure shows a weekly status of command latency.



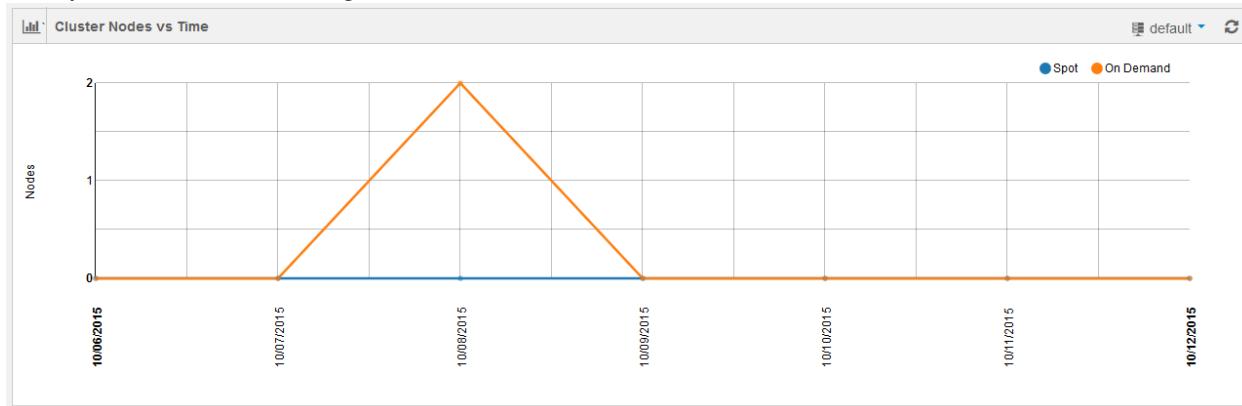
Command Error Distribution Status

The command errors' distribution is shown as a line chart with number of command errors in an account plotted against time. Time is dynamic and it is in minutes for hourly status and in days for weekly/monthly/custom status. The following figure shows a weekly status of command errors.



Cluster Nodes vs Time Status

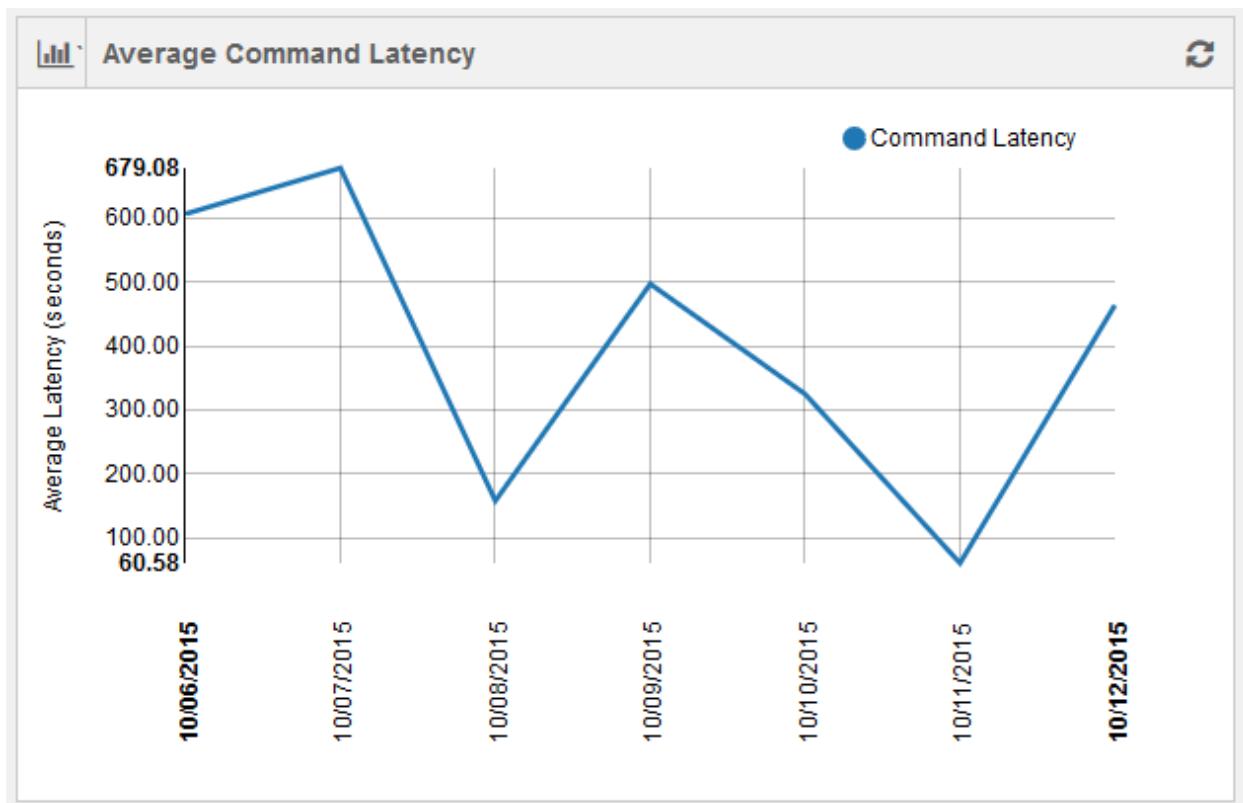
This line chart shows the usage of cluster nodes per account against time. Time is dynamic and it is in minutes for hourly status and in days for weekly/monthly/custom status. The following figure shows a graphical representation of weekly status of cluster nodes against time.



Two different line charts are shown for On-Demand cluster nodes and Spot cluster nodes.

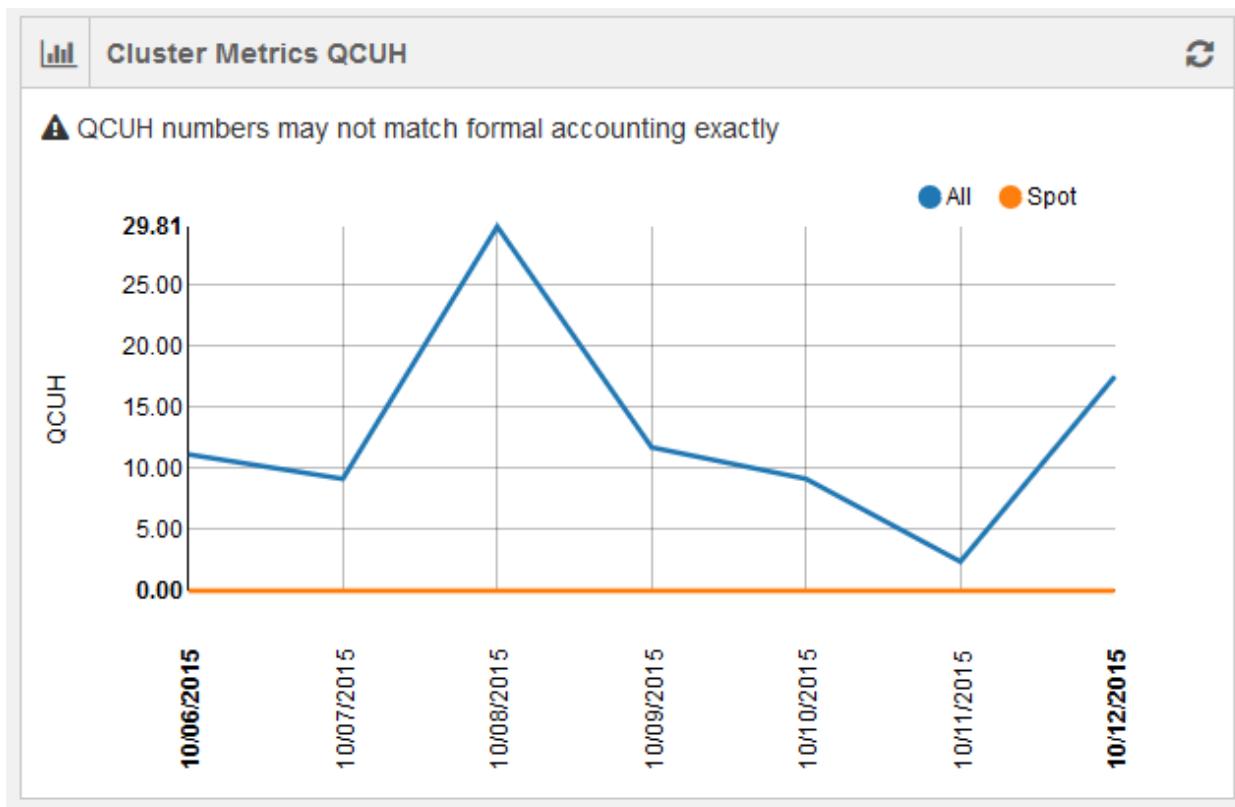
Average Command Latency Status

The average command latency is shown as a line chart with average command latency per account plotted against time. Time is dynamic and it is in minutes for hourly status and in days for weekly/monthly/custom status. The following figure shows a graphical representation of weekly average command latency.



Cluster Metrics QCUH Status

The cluster metrics is shown as a line chart of Qubole Compute Unit Hour (QCUH) per account plotted against time. Time is dynamic and it is in minutes for hourly status and in days for weekly/monthly/custom status. However, QCUH numbers in this chart may not match the formal accounting exactly. The following figure shows a graphical representation of weekly status of QCUH against time.

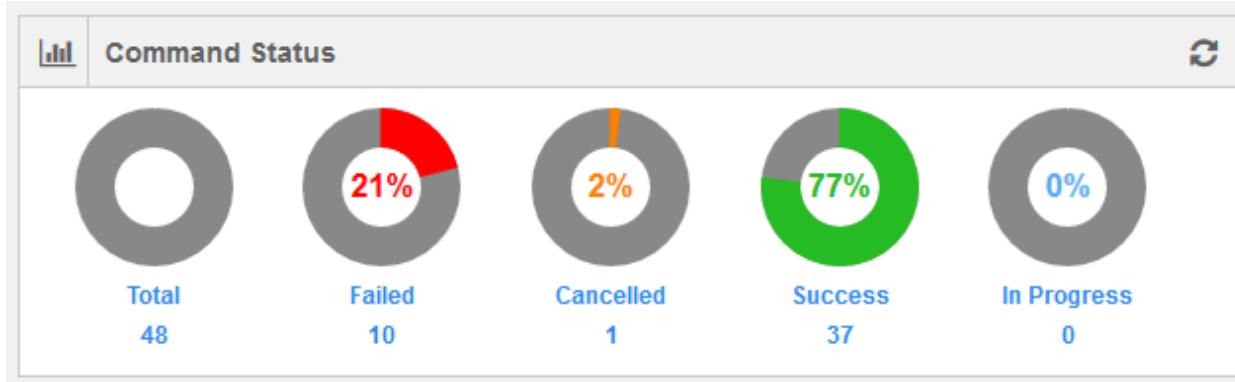


Two separate line charts are shown for spot nodes and all nodes.

Command Status

This chart gives the number of commands with **Failed/Cancelled/Success/In Progress** status along with total number of commands in an account as per the selected frequency that is hourly/daily/weekly/monthly.

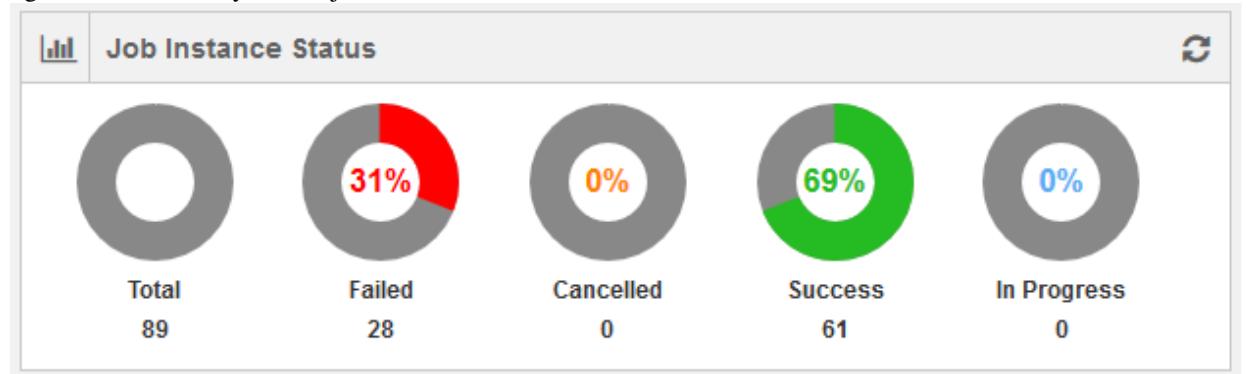
The following figure shows a weekly data of commands' status.



Job Instance Status

This chart gives the number of job instances with **Failed/Cancelled/Success/In Progress** status along with total number of job instances in an account as per the selected frequency that is hourly/daily/weekly/monthly. The following

figure shows a weekly data of job instances' status.



Longest Command/Job Instances Status

This chart displays commands and job instances with longest duration per account in a table during the selected frequency that is hourly/daily/weekly/monthly. By default, **Commands** are displayed. Click **Job Instances** to see them.

Longest Commands/Job Instances				
		Commands	Job Instances	
Id		Type	Command	Duration
12320636		Workflow	Workflow details not availa...	00:06:57
12149998		Hive Query	use datalogix; select count ...	00:04:32
12108557		Hive Query	use datalogix; set hive.cli.p...	00:04:05
12115546		Hive Query	use datalogix; set hive.cli.p...	00:02:25
12115891		Hive Query	use datalogix; set hive.cli.p...	00:02:07
12115124		Data Import	Mode: Advanced, Hive Tabl...	00:01:53
12115910		Hive Query	select c.name as account_...	00:01:37
12188250		Hive Query	show tables	00:01:34
12108391		Data Import	Mode: Advanced, Hive Tabl...	00:01:32
12115743		Hive Query	use datalogix; select * from...	00:01:29

Top Most Failing Active Jobs Status

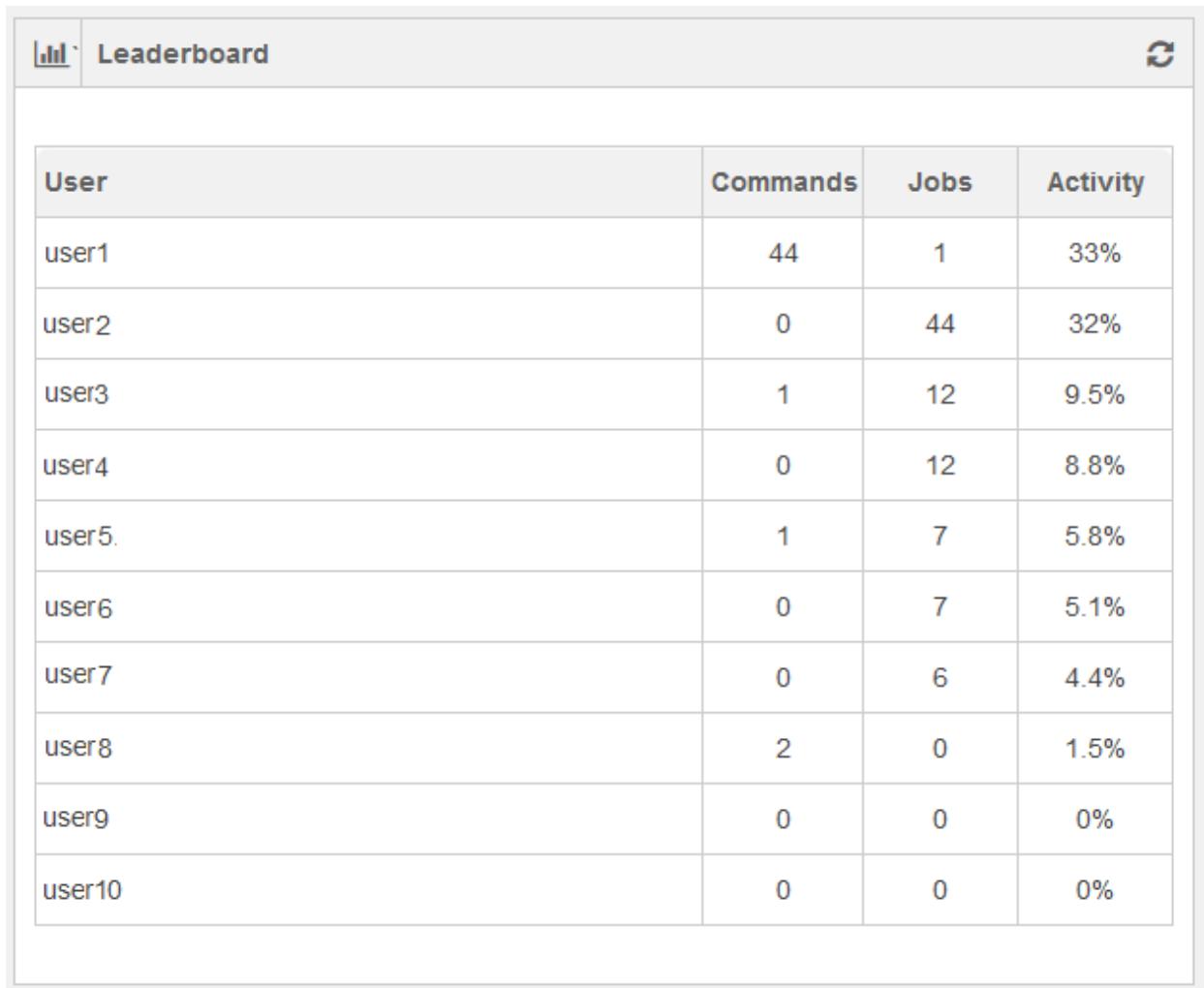
The following figure provides a list of top-most active jobs with failed job instances per account as per the selected frequency that is hourly/daily/weekly/monthly.

The screenshot shows a dashboard titled "Top Most Failing Active Jobs". The table has columns for Id, Type, Recent Instances, Frequency, and User. Each row shows a job ID, its type (Workflow or Hive Query), a visual representation of recent instances (green for successful, red for failed), the frequency of failure (e.g., 1440 minutes, 1 day), and the user who submitted it. The visual representation uses colored dots where green indicates success and red indicates failure.

Id	Type	Recent Instances	Frequency	User
596	Workflow	●●●●●●●●	1440 minutes	user1
250	Workflow	●●●●●●●●	1 days	user1
2390	Workflow	●●●●●●●●●●●●	1 days	user3
1551	Workflow	●●●●●●●●●●●●	1 days	user1
1552	Workflow	●●●●●●●●●●	1 days	user8
2277	Workflow	●●●●●●●●●●	1 days	user3
1553	Workflow	●●●●●●●●●●	1 days	user1
841	Hive Query	●●●●●●●●●●	1 days	user6
2447	Hive Query	●●●●●●●●●●	1 days	user5
3563	Workflow	●●●●●●●●●●	1 days	user8

Leaderboard Status

The following figure provides the list of user activity of an account as per the selected frequency that is hourly/daily/weekly/monthly.

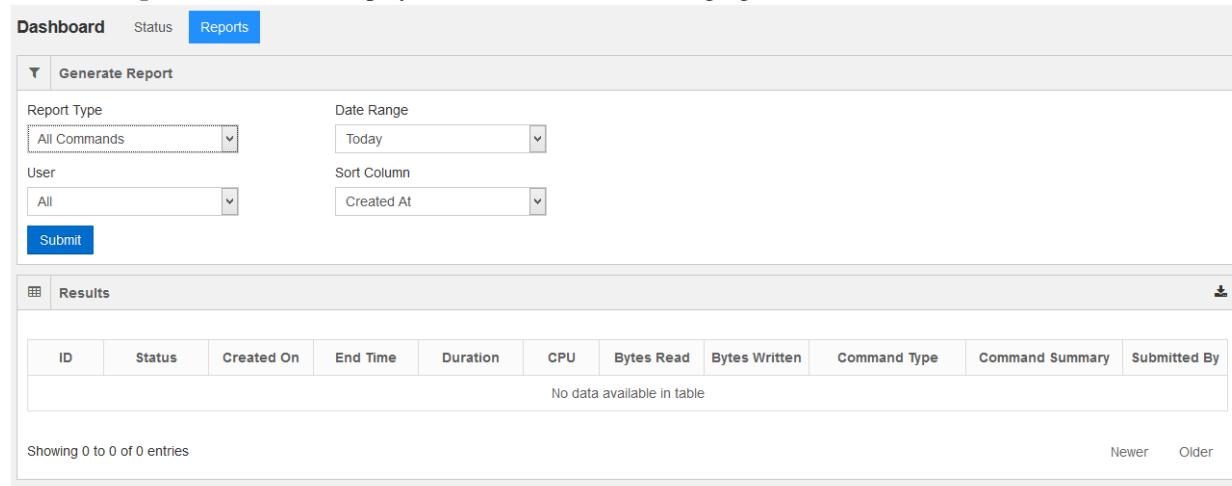


The screenshot shows a table titled "Leaderboard" with four columns: "User", "Commands", "Jobs", and "Activity". The data is as follows:

User	Commands	Jobs	Activity
user1	44	1	33%
user2	0	44	32%
user3	1	12	9.5%
user4	0	12	8.8%
user5	1	7	5.8%
user6	0	7	5.1%
user7	0	6	4.4%
user8	2	0	1.5%
user9	0	0	0%
user10	0	0	0%

Generating Reports using QDS Dashboard

In the QDS [Usage Dashboard](#), the **Reports** tab is used to generate reports such as cluster metrics and cluster usage. Click the **Reports** tab and it is displayed as shown in the following figure.



The screenshot shows the "Reports" tab of the QDS Usage Dashboard. At the top, there is a "Generate Report" section with dropdown menus for "Report Type" (set to "All Commands") and "Date Range" (set to "Today"), and dropdowns for "User" (set to "All") and "Sort Column" (set to "Created At"). A "Submit" button is located at the bottom of this section. Below this is a "Results" section containing a table with the following columns: ID, Status, Created On, End Time, Duration, CPU, Bytes Read, Bytes Written, Command Type, Command Summary, and Submitted By. The table currently displays the message "No data available in table". At the bottom of the "Results" section, it says "Showing 0 to 0 of 0 entries" and has "Newer" and "Older" links.

The following filters can be used to generate reports.

- **Report Type** - This filter is used to select the type of report. By default, it contains **All Commands**. The options available in the drop-down list are:
 - **Canonical Hive**
 - **Cluster Usage**
 - **Cluster Nodes**
- **User** - By default, the drop-down list contains **All** and the other option in the drop-down list is **Current**.
- **Date Range**- This option can be used to select a date range for a report. By default, the drop-down list contains **Today** and other options available in the drop-down list are:
 - **This Week**
 - **Last Week**
 - **This Month**
 - **Custom**. Selecting it gives an option to select a date range.
- **Sort Column** - Use this option to sort the columns in the report. By default, the drop-down list contains **Created At**. The other options available in the drop-down list are:
 - **CPU**
 - **Bytes Read**
 - **Bytes Written**

A sample all-commands report is as shown in the following figure.

Results											
ID	Status	Created On	End Time	Duration	CPU	Bytes Read	Bytes Written	Command Type	Command Summary	Submitted By	
12517276	error	2015-10-16 05:32:51	2015-10-16 05:33:39	a minute	-	-	-	CompositeCommand	HiveCommand: show t...	user1@qubo...	
12517305	done	2015-10-16 05:33:31	2015-10-16 05:34:57	a minute	-	-	-	HiveCommand	show tables;	user3@qubo...	
12517565	error	2015-10-16 05:39:57	2015-10-16 05:40:25	a few seconds	-	-	-	CompositeCommand	HiveCommand: show t...	user2@qubo...	
12517803	done	2015-10-16 05:45:57	2015-10-16 05:51:36	6 minutes	-	-	-	CompositeCommand	DblImportCommand: M...	user3@qubo...	
12517983	done	2015-10-16 05:50:40	2015-10-16 06:02:24	12 minutes	4060	2633054	2636453	CompositeCommand	HiveCommand: create...	user2@qubo...	
12526542	done	2015-10-16 09:30:02	2015-10-16 09:51:20	21 minutes	-	-	-	CompositeCommand	DblImportCommand: M...	user1@qubo...	
12526544	done	2015-10-16 09:30:02	2015-10-16 09:53:23	23 minutes	-	-	-	CompositeCommand	DblImportCommand: M...	user7@qubo...	
12526558	done	2015-10-16 09:30:04	2015-10-16 09:46:36	17 minutes	23320	1522678	1558480	CompositeCommand	HiveCommand: create...	user1@qubo...	
12526748	done	2015-10-16 09:33:56	2015-10-16 09:52:39	19 minutes	-	-	-	CompositeCommand	DblImportCommand: M...	user7@qubo...	
12530302	done	2015-10-16 11:09:59	2015-10-16 11:22:38	13 minutes	3880	1363204	1365514	CompositeCommand	HiveCommand: create...	user1@qubo...	

Newer Older

The report displays the most-recent commands are shown in the first page and it contains the **Newer** button disabled. Click the **Older** button to see the next set of commands that are older than the recent commands.

2.2.8 Data Store

It is often useful to import or export data to data stores other than S3. For example, one may want to run a periodic Hive query to summarize a dataset and export the results into MySQL database. Alternatively, one can import data from a relational database into S3/Hive periodically. A data store identifies an external end point to import/export data from QDS. Currently QDS supports the following as data stores:

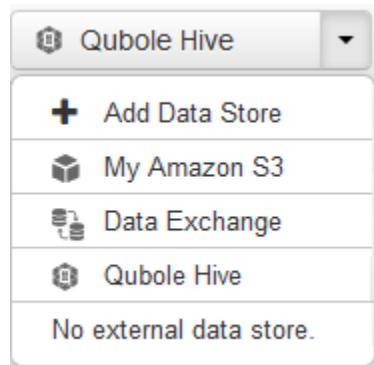
- MySQL
- Postgres
- Vertica
- Redshift
- Mongo Db
- Microsoft SQL Server

Qubole is actively working on supporting more databases. The usage of a data store is described in more detail in the [Data Import](#) and [Data Export](#) sections.

Creating a Data Store

Create a new data store at [Explore](#) by performing the following steps:

1. Pull down the drop-down list to see it as shown in the following figure.



Only a system administrator can add a data store and hence, see the Add Data Store option.

By default, it shows **Database Type** as **MySQL** as shown in the following figure.

Connection Details

Database Type: MySQL

Database Name: Database Name

Server Address: Server Address

Port: Port (blank for default)

Username: Username

Password: Password

Region: us-east-1 (North Virginia)

Save **Reset** **Cancel**

* Please ensure that our Amazon account id (805246085872) and our security group ('default') have access privileges to your database.
 - If your database is not in VPC then you can give these permissions by adding rules to security group of database.
 - If your database is in VPC use 'on-premise' as database region.
 See [here](#) for more information.

2. Select the required database type in the **Database Type**.
3. Enter the name in the **Database Name** text field. This is the data store name that will be created in the QDS.
4. Enter the server address in the **Server Address** text field.
5. Enter the port number in the **Port** text field. You can leave this text field blank for default port numbers.
6. Enter the username of the host in the **Username** text field.
7. Enter the password of the host in the **Password** text field.
8. Select the corresponding **AWS Region**. See *Firewall Configuration for QDS access to Database* for more information.

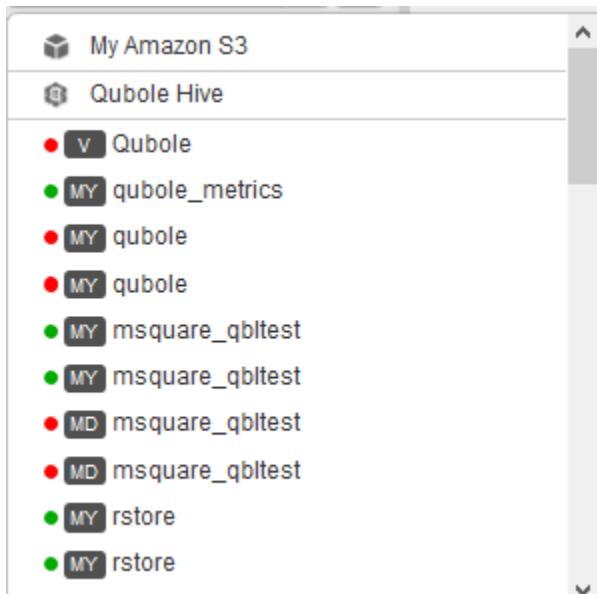
An appropriate error message is displayed if QDS fails to connect to the database.

When database is not in one of the AWS region supported by Qubole, select on-premise as the db location.

Click **Save** to add the data store. Click **Reset** to reset the values. Click **Cancel** to not add the data store.

The data store is marked activated if QDS gets connected to the database. You can check the list of data stores in the drop-down list in the [Explore](#) page.

The activated data stores contain a green dot against them. Non-activated data stores contain a red dot against them as illustrated in the following figure.



See [Create a DbTap](#) for more information on creating a data store using the REST API.

Firewall Configuration for QDS access to Database

QDS supports the following AWS regions:

- us-east-1
- us-west-2
- eu-west-1
- ap-singapore-1

1. If the database is a RDS instance, then the database security group to which the RDS instance belongs must grant access privileges to the QDS account/default security group pair. See [RDS FAQ](#) for reference.
2. If the database is running on an EC2 machine, then the security group to which the EC2 instance belongs, must grant access to QDS account/default security group pair. See [EC2 User Guide](#) for reference.

If the database is not located in the AWS regions supported by Qubole or is located outside AWS, then the database must have its network ports open to everyone and use username/password based security to grant access to Qubole. **When database is not in any AWS region supported by Qubole, then select on-premise as the database location.**

SSL support for QDS access to Postgres Database

You can use SSL based access to the Postgres database if you configure the database to use [SSL](#). While using SSL, QDS expects you to open database ports to everyone as SSL takes care of the QDS security. If the Postgres database is in *us-east-1*, you can continue with opening database ports to only QDS.

2.2.9 Data Export

Export data from S3 and Hive tables to relational databases such as MySQL, Postgres, Vertica and Redshift. QDS uses [Sqoop](#) to export data. First, create a [Data Store](#) and set it as the destination.

Controls are grouped into two extraction modes:

1. Directory Export

Specify which S3 directory to export to the data store destination. Note that this is a **non recursive** export. You can specify the Field Delimiter (default is ^A).

2. Hive Table Export

Specify which Hive table to export to the data store destination. QDS supports exporting all Hive table formats, namely text serde, ORC (Optimized Row Columnar File), and RC (Record Columnar File) formats.

Tables with No Partitions

If the Hive table has no partitions, then all data in the Hive table is exported.

Tables with Partitions

If the Hive table has partitions, then you need to specify a value for each partition in the **partition spec** field. Example: If the Hive table has partitions ‘date’ and ‘country’ with values ‘2012-01-01’ and ‘USA’ respectively, then the partition spec must be specified as: date=2012-01-01/country=USA. The Data Export command gives a relevant error if you only specify date=2012-01-01 or country=USA as the partition spec. Refer to *Support for Redshift* for specific details.

Database Insertion Modes

QDS supports *append* and *update* modes for all the supported databases as well as *upsert* mode for MySQL. In the **Update** mode, you must specify the update keys also as they are used by the update query.

Redshift Support

For Redshift data stores, use the Copy command (as recommended by AWS) to export data from S3 to Redshift. This does not work if the Redshift Database and S3 location are in different regions. Qubole automatically falls back to normal insert queries in case the copy command fails but this can be significantly slower. Also, this means that while doing Hive Table Export, it is not possible to export the partition columns via **Copy** command because partition columns are not part of the data. However, fall back to insert-based exports as the **Copy** command fails because of schema mismatch.

Extending the Data Export Command

QDS provides the ability to build more complex systems on top of the existing functionality. You can export custom columns, custom serdes, or multiple partitions. Let us take an example where you want to export only a few columns from a Hive table or you want to export many partitions in single command:

- Select a Workflow in the **Compose** Tab on **Analyzeze**.
- Select Hive command as the first sub-command of the workflow. Write the custom Hive query whose result you want to export. You can write this query either as *insert overwrite directory s3://...* or as *create table as ...*
- Select Data Export as the second sub-command of the Workflow. Use *Hive Table Export* or *Directory export* depending on whether you used *create table* or *insert overwrite directory* in the previous step.
- Optionally select Hive command as the last sub-command of the Workflow to cleanup S3 using a *drop table* command.

2.2.10 Data Import

Import data from relational databases such as MySql, Postgres, Vertica and Redshift as well as non-relational databases such as MongoDB. QDS uses [Sqoop](#) to import data. First, create a *Data Store* and set it as a source.

Controls are grouped into two extraction modes:

- Simple
- Advanced

Simple

In this mode, one can specify a table name (present in the DbTap), list of columns, and a where clause.

Advanced

In this mode, a free form query can be specified. For example, the user can even join multiple tables and extract the (denormalized) data. However, if parallel extracts are desired, then a ‘\$CONDITIONS’ macro must be part of a where clause in the query (see below).

Controlling Parallelism

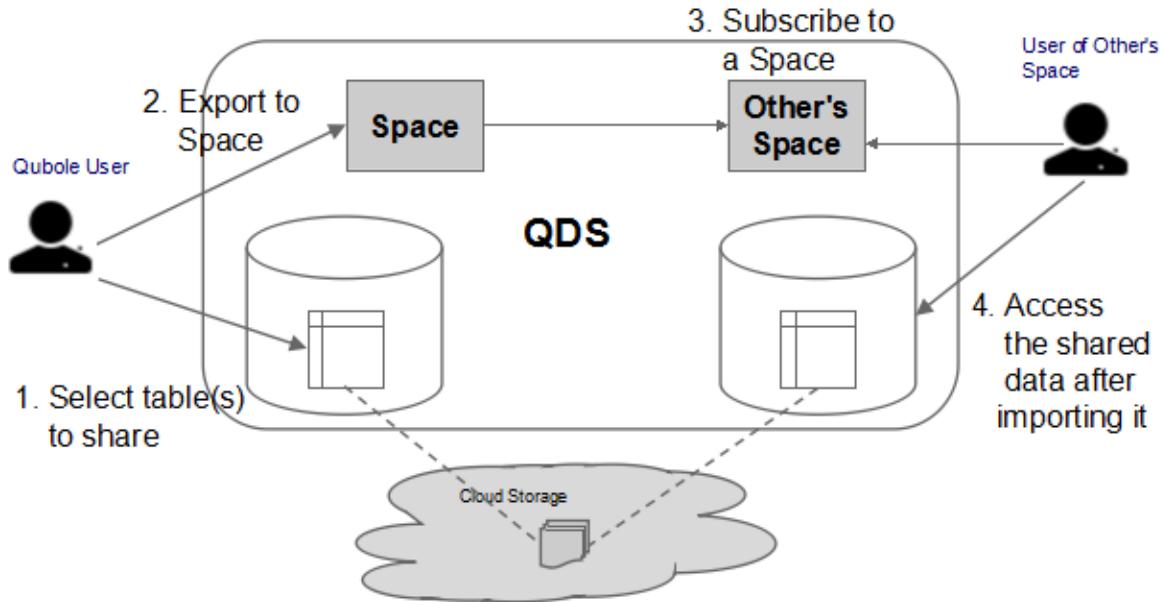
Sqoop can parallelize database extracts. By default, it automatically figures out the level of parallelism and perform up to 4 extracts in parallel. However, the user can specify the level of parallelism explicitly. Most importantly, the level of Parallelism can be set to 1 and this disables parallel extracts.

In the *Simple* mode, Sqoop tries to parallelize extracts using an integer primary key (if one is available). In the *Advanced* mode, the user has more control over specifying how to parallelize the extracts. A user can specify a column to parallelize on (this is the column that must be visible in the scope of the ‘\$CONDITIONS’ macro embedded inside the free form query). Finally, the user must specify another query called the Boundary Query that returns the minimum and the maximum of the column to parallelize on. Qubole strongly encourages reading through the sqoop documentation on *Free Form Imports* to understand these controls.

2.2.11 Data Exchange

Qubole has introduced Data Exchange, a feature that enables an organization to share data and analytics with partners. With this feature, you can still retain complete control over access, permissions, and usage rights while granting access to the partners or members of other organizations. The following figure shows a high-level overview of Data Exchange in QDS.

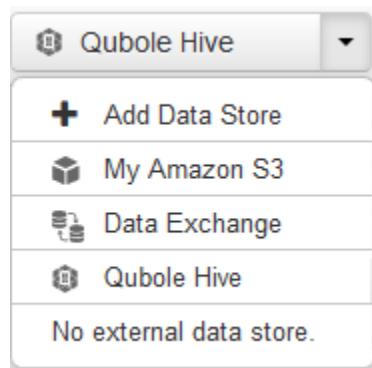
Data Exchange Workflow



As an overview, the table data can be shared into a Space. A partner can subscribe to a Space by providing access details for that Space. Subscribed Spaces are listed in Other's Space section of the interface. A partner can then navigate to the subscribed Space to see exported tables and import those tables into his account and access the Hive table data to do further analysis.

QDS has added Data Exchange as an option in [Explore](#) along with Qubole Hive, Amazon S3, and data stores.

Navigate to [Explore](#) and pull down the drop-down list to see it as shown in the following figure.



The following topics describe how to use the QDS user interface to enable Data Exchange:

- [Adding Data Exchange Spaces](#)
- [Managing My Spaces](#)
- [Managing Other's Spaces](#)

2.2.12 Query Export

A common operation is to export the results of a query to a database. This operation can be modelled as a workflow of Hive Query followed by a Data Export. Query Export is a template with fields to fill in the Hive Query and the Data Export Command. The Hive Query should write its results out to a pre-determined location in a S3 bucket. The Data Export command should export from the pre-determined location to a DbTap of your choice.

Example

(only a subset of fields are shown)

Hive Query

```
insert overwrite directory 's3://..../daily_tick_data/temp'
select
  stock_symbol,
  max(high),
  min(low),
  sum(volume)
from
  daily_tick_data
where
  date1='2011-07-01'
group by
  stock_symbol
```

Data Export

```
Export Dir: s3://..../daily_tick_data/temp
```

2.2.13 Refresh Table

Hive partitions are often backed by directories in S3 and are only created when the ‘recover partitions’ command is run. Refresh Table template can create such partitions.

It is important to understand the concept of Stability Interval in the context of the Refresh Table template. Directories in S3 are typically created first – and then data is uploaded into it. The Hive partition corresponding to such a directory should not be created until the process of populating the directory is complete. The Stability Interval defines the number of minutes that must elapse since the last modification to a directory – before the Hive partition corresponding to it is created.

As an example consider a process where data is loaded into a S3 directory at midnight every day. The process of uploading the logs takes about an hour. So the Refresh Table template can be configured with a stability interval of 2 hours (120 minutes) to make sure that the process of uploading the data is complete before the Hive partitions are created.

2.2.14 Schedule

About the Schedule UI Interface

Qubole Scheduler provides a way to run commands at specific intervals without manual intervention. Navigate to the **Schedule** tab to see the periodic jobs. The list of jobs is as shown in the following figure.

The screenshot shows the Qubole Schedule UI. On the left, there is a table listing periodic jobs. The columns are Name, Type, Frequency, Status, and Recent. The table contains the following data:

Name	Type	Frequency	Status	Recent
596	Workflow	1440 minutes	Active	
250	Workflow	1 days	Active	
205	Data Import	1440 minutes	Active	
200	Hive Query	14 days	Active	

At the top of the table area, there is a search bar with the query "status:Active && user_id>All && type>All" and a "Filter" button. Below the table is a "Show More" button. On the right side of the interface, there is a message: "Please select a job to see the details".

Click **Show More** to see the complete list of periodic jobs. See *Qubole Scheduler* for more information.

Viewing a Job

In the **Schedule** tab, select the listed job that you want to see. After you select a job from the list, the job details are visible in the **Schedule** tab as shown in the following figure.

The screenshot shows the Qubole Schedule UI. On the left, there is a table listing periodic jobs, identical to the one in the previous figure. On the right, there is a detailed view of a selected job. The job details include:

- Command**: Hive Query
- Hive Statement**:

```
1 show tables;
```
- Actions**: A set of icons for managing the job, including Active, Stop, Delete, Edit, and Refresh.
- Show instances**: A link to view the instances of the job.

Alternatively, if you know the ID/name of the job that you want to see, you can use **Filter**. See *Filtering a Job* for more information.

Scroll down to see the complete details of a job as shown in the following figure.

Active Show instances

Command

Hive Query

Hive Statement

```
1 show tables;
```

Cluster Label

default

Tags

Schedule

Next instance at	2015-05-13 00:00 UTC
Start Time	2013-05-15 00:00 UTC
End Time	2037-05-15 00:00 UTC
Periodicity	14 days



Click the **edit icon** for editing a job.



Click the **stop icon** for stopping a job.



Click the **suspend icon** for suspending a job.

A dialog to suspend with **OK** and **Cancel** buttons is displayed.

Click **OK** to suspend a job. In the job details page, the suspended state is shown and the suspend icon replaced with a resume icon as shown in the following figure.

Suspended

Show instances

Command

Note: The default filter shows only running jobs. When a job is suspended, it disappears from the list of running jobs. To see the list of suspended jobs, select the status as suspended. See [Filtering a Job](#) for more information.



You can resume a suspended job any time by clicking the **resume icon** .

A dialog to resume with **OK** and **Cancel** buttons is displayed. Click **OK** to resume a job.



Click the **clone icon** for cloning a job.



While cloning a job, you can retain other details but you must rename the schedule.

Click the **permalink icon** to see the job's permalink.



Note: Use the tooltip to know more information on each field.

Here are the fields related to a periodic job:

- The **Active** label indicates that the job is active. The query details are mentioned in the **Command** text field.
- The **Show Instances** button when clicked displays the instances of a periodic job. See [Viewing a Job Instance](#) for more information.
- **Cluster Label** provides the labels of a cluster on which a specific periodic job is run.
- The **Tags** text field displays a list of specified tags (if any) that are used to group commands together. Tags help in identifying commands. A maximum of six tags can be added. Each tag can contain a maximum of 20 characters.
- **Schedule** displays the next instance, start time/end time, and the periodicity (frequency) of a periodic job.

Note: If macros are set while creating a periodic job, then **Macros** are displayed above **Schedule**.

- **Concurrency** shows the number of concurrent jobs that are allowed.
- **Skip Missed Instances** provides an option to select it, that skips instances supposed to have run in the past.

- **Dependencies** has three options to be set for a job:
 - **No Dependency** (selected by default)
 - **Wait for Hive Partition**. See [Configuring Hive Tables Data Dependency](#) for more information.
 - **Wait for S3 Files**. See [Configuring S3 Files Data Dependency](#) for more information.
- **Notification** is an optional field to be selected if you want to be notified through email on an instance failure.

Viewing a Job Instance

When you view a job, on the top-left corner, you see **Show Instances**, which when clicked displays the list of instances of a particular job as shown in the following figure.

Job Name 200 ✓

Job Details

Instances

status: All			Filter 🔍
Id	Nominal Time	Created At	
52	2015-04-29T00:00:00Z	2015-04-29T00:00:01Z	🕒
51	2015-04-15T00:00:00Z	2015-04-15T00:00:04Z	🕒
50	2015-04-01T00:00:00Z	2015-04-01T00:00:00Z	🕒
49	2015-03-18T00:00:00Z	2015-03-18T00:00:08Z	🕒
48	2015-03-04T00:00:00Z	2015-03-04T00:00:02Z	🕒
47	2015-02-18T00:00:00Z	2015-02-18T00:00:09Z	🕒
46	2015-02-04T00:00:00Z	2015-02-04T00:00:02Z	🕒
45	2015-01-21T00:00:00Z	2015-01-21T00:00:08Z	🕒
44	2015-01-07T00:00:00Z	2015-01-07T00:00:00Z	🕒
43	2014-12-24T00:00:00Z	2014-12-24T00:00:06Z	🕒
42	2014-12-10T00:00:00Z	2014-12-10T00:00:02Z	🕒
41	2014-11-26T00:00:00Z	2014-11-26T00:00:01Z	🕒
40	2014-11-12T00:00:00Z	2014-11-12T00:00:06Z	🕒
39	2014-10-29T00:00:00Z	2014-10-29T00:00:05Z	🕒
38	2014-10-15T00:00:00Z	2014-10-15T00:00:08Z	🕒

Show More

Please select an instance to see the details

Select an instance to see the details of that instance as shown in the following figure.

The screenshot shows the Qubole Data Service interface for a job named "Job Name 200".

- Job Details:** Shows the status as "All".
- Instances:** A table listing 37 instances. The columns are "Id", "Nominal Time", and "Created At". The first instance has the ID 52, Nominal Time 2015-04-29T00:00:00Z, and Created At 2015-04-29T00:00:01Z.
- Command:** A tab showing a "Hive Query" section with the statement "show tables;".
- Log:** A tab showing the output of the query execution, including file processing logs and execution statistics: "Processing file s3://data/com/logs/production/scripts/bootstrap", "Processing file /tmp/mdhist20150429-30066-rp2yj5-0", "OK", and "Time taken: 1.042 seconds, Fetched: 582 row(s)".

The query is seen in the **Command** tab. The **Log** tab contains the query logs.

Filtering a Job

In the **Schedule** tab, click the **Filter** button. The **Filter Jobs** dialog is displayed. Enter the job name (or ID) and select the status, user, and command-type from the corresponding drop-down lists (default selection is **All** for all drop-down lists) as illustrated in the following figure.

The screenshot shows a search bar at the top with the query "name:250 && status>All && user_id>All && type>All". To the right of the search bar is a "Filter" button with a magnifying glass icon and a dropdown arrow. Below the search bar is a modal dialog titled "Filter Jobs". The dialog contains five input fields with dropdown arrows: "Job Id" (Job Id), "Job Name" (250), "Status" (All), "User" (All), and "Command Type" (All). At the bottom of the dialog are three buttons: "Apply" (in blue), "Clear" (in blue), and a close button (an 'X').

Click **Apply** to get that specific job listed. Click **Clear** to reenter the details.

Creating a New Job

Click **New Job** in the **Schedule** tab. **Create Job** is displayed as shown in the following figure.

Create Job

Schedule Name

Command i

Hive Query▼

Hive Statement ▾

1

Cluster Label

default ▾

Tags i

Macros i

Have you used macros in the command? Click [+ Add Macro](#) to add macros.

Schedule

Start Time i

2015 ▾ May ▾ 2 ▾ — 00 ▾ : 00 ▾

End Time i

2018 ▾ May ▾ 2 ▾ — 00 ▾ : 00 ▾

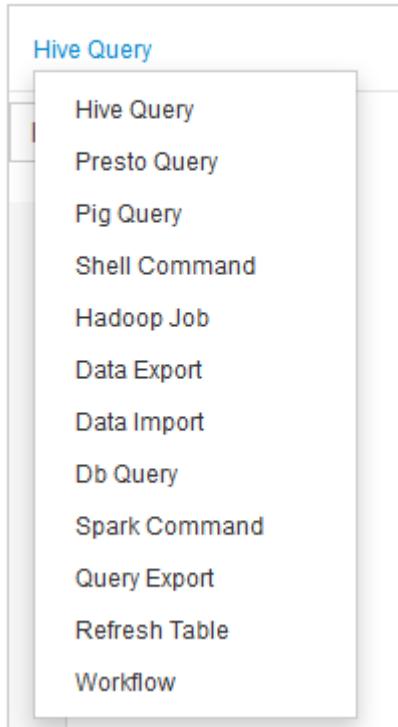
Time Zone i

(GMT-10:00) Hawaii ▾

Perform the following steps to create a job:

1. Enter a name in the **Schedule Name** text field. This field is optional. If it is left blank, a system-generated ID is set as the schedule name.
2. In the **Command** field, select the type of query from the drop-down list, and enter the query in the text field. The following types of queries are supported by Qubole Scheduler.

Command



See [Analyze](#) for more information on composing different types of queries.

If the query statement is in an S3 location, select **S3 Script Location** from the drop-down list that contains **<CommandType> Statement** selected by default. Enter the S3 location that contains the script. Qubole now allows setting an S3 script location for Presto queries.

3. The **default** label is selected by default in the **Cluster Label** list. Select a different cluster label from the drop-down list.
4. In the **Tags** text field, add one or a maximum of six tags to group commands together. Tags help in identifying commands. Each tag can contain a maximum of 20 characters. It is an optional field.
5. If you have used macros in the query, click the **+Add Macro** button available in the **Macros** field. Else, proceed to the next step. After you click the **+Add Macro** button, the macros are displayed as shown in the following figure.

Macros



The image shows a form for adding macros. It has two input fields: "Variable" and "Expression", separated by an equals sign (=). Below the input fields are two buttons: "+Add Macro" (in blue) and "Validate Macros" (in green).

Enter the variable and expression and click **Validate Macros** for validating it. See [Macros in Scheduler](#) for more

information. Click **+Add Macro** to add another macro. Else, proceed to the next step.

Note: Use the tooltip  to know more information on each field.

6. In the **Schedule** field, set:

- (a) **Start Time** by selecting the year, month, date, hour and minute from the corresponding drop-down lists.
 - (b) **End Time** by selecting the year, month, date, hour and minute from the corresponding drop-down lists.
 - (c) **Time Zone** by selecting the appropriate timezone from the drop-down list.
 - (d) **Periodicity** by entering a value and selecting the periodicity type from the drop-down list. **Minutes** is the default periodicity type.
7. Select the number of concurrent jobs allowed from the **Concurrency** drop-down list if you do not want the default value.
8. Select **Skip Missed Instances** if you want to skip instances supposed to have run in the past. By default, this option is unselected. When a new schedule is created, the scheduler runs instances from start time to the current time. For example, if a daily schedule is created from Jan 1 2015 on May 1 2015, jobs are run for Jan 1 2015, Jan 2 2015, and so on. If you do not want the scheduler to run the missed instances for months earlier to May, select the checkbox to skip them.
9. **Dependencies** has three options to be set for a job:
- **No Dependency** (selected by default)
 - **Wait for Hive Partition**. See [Configuring Hive Tables Data Dependency](#) for more information.
 - **Wait for S3 Files**. See [Configuring S3 Files Data Dependency](#) for more information.
10. **Notification** is an optional field to be selected if you want to be notified through email about instance failure. Once you select the **Send email notifications** checkbox, **Email Type**, **Email List**, and **Event** are displayed as shown in the following figure.

Notification (optional)

Send email notifications

Email Type:

Immediate

▼

Email List:

qubole@qubole.com

x

Event:

Failure Success

Select the **Email Type** option, **Daily digest** to receive daily digests if a job periodicity is in minutes or hours. The default email type is **Immediate**.

By default, the current user's email ID is added in the **Email List** field. You can add additional emails as required.

By default, in **Events**, **Failure** and **Success** type of events are not selected. You can select both type of events or any one of them.

11. Click **Submit** to add a new periodic job after you are done with filling the required details. Click **Cancel** if you do not want to submit a job.

Configuring S3 Files Data Dependency

S3 files' dependency implies that a job runs if the data is arrived in S3 buckets. You can schedule a job to run at a specific date and time, either once or on a repetitive basis if the data exists. You can define repeat intervals such as last 6 hours, last 6 days, last 3 weeks, and last 7 months.

To schedule jobs at periodic intervals, Qubole Scheduler requires the following information:

- Start day or time (parameter: window_start)
- End day or time (parameter: window_end)
- Day or time interval that denotes when and how often data is generated (parameter: interval)
- Nominal time which is the logical start time of an instance

The following table shows how to create data in S3 files for the previous day's data with daily interval.

Sequence ID	Nominal Time	Created At	Dependency
1	2015-01-01 00:00:00	2015-04-22 10:00:00	s3://.../2014/12/31/...
2	2015-01-02 00:00:00	2015-04-22 10:15:00	s3://.../2015/01/01/...
3	2015-01-03 00:00:00	2015-04-22 10:30:00	s3://.../2015/01/02/...

To configure S3 files dependency, select the **Wait For S3 Files** option available in **Dependencies**. The S3 Files fields are displayed as shown in the following figure.

Dependencies

No Dependency Wait for Hive Partition Wait for S3 Files

File 1 X

Location eg: s3://bucket.domain.com/data/log_%Y%m%c

Window Start eg: -2

Window End eg: 0

+ Add More

Time out eg 10 in Minutes

Note: Use the tooltip to know more information on each field or checkbox.

The following steps explain how to set S3 File dependency:

1. Enter the S3 location in the format: S3://.../%Y/%M/%d/%h/%m....
2. **Window Start** and **Window End** defines the range of interval to wait for. The values are integers in units of time, hour/day/week/month/year.

Enter the **Window Start** value. See [Hive Datasets as Schedule Dependency](#) for more information on the window start parameter. An instance runs waits for files for the specified time range. **Window Start** specifies the start of this range. For example if you set -1 as window start time that implies 1 hour before/previous day/week/month/year. If it is 2 hour/day/week/month/year before, the value of window start is -2 and so on.

3. Enter the **Window End** value. See [Hive Datasets as Schedule Dependency](#) for more information on the window end parameter. An instance runs waits for files for the specified time range. **Window End** specifies the end of this range. For example, if the interval is for 7 days and window start value is -6, the window end time is 0.

The value 0 implies now, -1 implies 1 day ago, and -2 implies 2 days ago. Correspondingly, for hourly/daily/weekly/monthly/yearly interval (frequency), the value 0 denotes now. -1 denotes 1 hour/day/week/month/year ago. -2 denotes 2 hour/day/week/month/year ago and so on.

Qubole Scheduler supports waiting for data. For example, waiting for 6 weeks of data implies that window_start is -5 and window_end is 0.

4. Enter **Time out** in minutes.

Note: When the data arrival interval and the scheduler interval are different, then the scheduler interval follows its own frequency to process the data. For example, if the data arrival interval is hourly and the scheduler interval is daily, the scheduler waits for an entire day's data.

Click **+Add More** to add a second file. Repeat steps 1-3 to enter the file details. Timeout is set only once as it is applicable to all files. The following figure illustrates how S3 file location and window start and window end values are set.

Dependencies i

No Dependency Wait for Hive Partition Wait for S3 Files

File 1



Location ?

s3://data.bucket.com/files/201503101200

Window Start ?

-6

Window End ?

0

File 2



Location ?

s3://data.bucket.com/files/201503091200

Window Start ?

-5

Window End ?

0

+ Add More

Time out ?

1

in Minutes

Click **+Add More** to add the number of files as per the periodicity/frequency of the schedule.

Configuring Hive Tables Data Dependency

To configure hive partitions dependency, select **Wait For Hive Partitions** option available in **Dependencies**. See *Hive Datasets as Schedule Dependency* for more information. The Hive Partition fields are displayed as shown in the following figure.

Dependencies

No Dependency Wait for Hive Partition Wait for S3 Files

Table Name  

Time out  in Minutes

Note: Use the tooltip  to know more information on each field or checkbox.

Select a table with Hive partitions. Once you select the table, the **Table Data** and **Dependency on Partition Range** are displayed as shown in the following figure.

Dependencies

No Dependency Wait for Hive Partition Wait for S3 Files

Table Name  

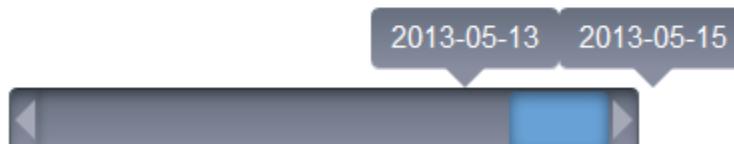
Table Data  Interval = 1 days, Partitions = month, month

Dependency on Partition Time Range

Select partition time range using 'start time' as an example launch time

For Launch Time Thursday, 2013-05-15 00:00

Partition Time Tuesday, 2013-05-13 00:00 to Wednesday, 2013-05-14 23:59
Range



Time out  in Minutes



Click the edit icon that is next to **Table Data**.

The window to enter table-specific data is to displayed as shown in the following figure.

Please enter table specific data for: default_qubole_memetracker

This table requires some additional data. Please configure below.

Interval: ?

Increment: ?

Partition Column "month" :

Specify DateTime Mask for this Partition?

Recent Values 2008-08, 2008-11, 2008-12, 2009-01, 2009-03, 2009-04

?

Date/Time Mask: ?

?

2008-08

Year Month Day Hour Minute

Partition Column "month" :

Specify DateTime Mask for this Partition?

Recent Values 2008-08, 2008-11, 2008-12, 2009-01, 2009-03, 2009-04

?

Date/Time Mask: ?

?

2008-08

Year Month Day Hour Minute

Set the **Interval** and select an incremental value from the **Increment** drop-down list. The default value is minutes.

In the **Partition Column**, select the **Specify DateTime Mask for this Partition** option if you want to specify it.

Set the date time mask in this format: YY-MM-DD-HH-MM as illustrated in the above figure.

Click **Save** after creating/modifying the table-specific data. Click **Close** to retain the existing settings.

Configure **Timeout** in minutes.

Dependency on Partition Range can be changed. Large data sets are typically divided into directories. Directories map to partitions in Hive. Currently, partitions in Hive are populated manually using the following command (picking up for the miniwikistats table):

```
ALTER TABLE miniwikistats RECOVER PARTITIONS;
```

Through the user interface, the partition range can be set. For example, if you want data for the last 9 days instead of last 3 days, then it can be changed as required by pulling the horizontal scrollbar (blue) and increase the range by expanding it as illustrated in the following figure.

Dependencies

No Dependency Wait for Hive Partition Wait for S3 Files

Table Name

Table Data Interval = 1 days, Partitions = month, month

Dependency on Partition Time Range

Select partition time range using 'start time' as an example launch time

For Launch Time Thursday, 2013-05-15 00:00

Partition Time Tuesday, 2013-05-06 00:00 to Wednesday, 2013-05-14 00:00

Range

2013-05-06 2013-05-14

◀ ▶

Time out in Minutes

Configure **Timeout** in minutes to change the default/Previously-set time.

Editing a Job

Navigate to the **Schedule** tab to see the periodic jobs. Select a job from the list to see its details. In the top of the job information page, click the **edit icon**  to modify the schedule.

The **Edit Schedule** page is displayed as illustrated in the following figure.

Edit Schedule

Schedule Name

Command i

Hive Query ▼

Hive Statement ▼

```
1 show tables;
```

Cluster Label

Tags i

Macros i

Have you used macros in the command? Click + Add Macro to add macros.

Schedule

Next instance at 2015-05-13 00:00 UTC

Start Time ?

2013 ▼ May ▼ 15 ▼ — 00 ▼ : 00 ▼

End Time ?

180037 ▼ May ▼ 15 ▼ — 00 ▼ : 00 ▼

Time Zone ?

(GMT+00:00) UTC ▼

Perform the following steps to edit a schedule:

1. Change the **Schedule Name** if you want to name it differently.

Follow the steps 2-11 as described in [Creating a New Job](#).

2.2.15 Example Datasets

Run the following SQL as a Hive query to get access to the TPC-DS scale 1000 dataset in ORC format. The tables are created in a hive database named `tpcds_orc_1000`. The largest table `tpcds_orc_1000.store_sales` is around 360 GB in uncompressed form. This table may be queried using Hive or Presto.

```
drop database if exists tpcds_orc_1000 cascade;
create database tpcds_orc_1000;
use tpcds_orc_1000;

create external table store_sales
(
    ss_sold_date_sk          int,
    ss_sold_time_sk          int,
    ss_item_sk                int,
    ss_customer_sk             int,
    ss_cdemo_sk                int,
    ss_hdemo_sk                int,
    ss_addr_sk                  int,
    ss_store_sk                int,
    ss_promo_sk                int,
    ss_ticket_number            int,
    ss_quantity                  int,
    ss_wholesale_cost           float,
    ss_list_price                 float,
    ss_sales_price                 float,
    ss_ext_discount_amt           float,
    ss_ext_sales_price           float,
    ss_ext_wholesale_cost           float,
    ss_ext_list_price             float,
    ss_ext_tax                     float,
    ss_coupon_amt                  float,
    ss_net_paid                   float,
    ss_net_paid_inc_tax           float,
    ss_net_profit                  float
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/store_sales';

create external table customer_demographics
(
    cd_demo_sk                  int,
    cd_gender                     string,
    cd_marital_status              string,
    cd_education_status             string,
    cd_purchase_estimate             int,
    cd_credit_rating                  string,
    cd_dep_count                    int,
    cd_dep_employed_count             int,
    cd_dep_college_count              int
)
```

```
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/customer_demographics';

create external table date_dim
(
    d_date_sk          int,
    d_date_id          string,
    d_date              timestamp,
    d_month_seq         int,
    d_week_seq          int,
    d_quarter_seq       int,
    d_year               int,
    d_dow                int,
    d_moy                int,
    d_dom                int,
    d_qoy                int,
    d_fy_year            int,
    d_fy_quarter_seq     int,
    d_fy_week_seq        int,
    d_day_name           string,
    d_quarter_name       string,
    d_holiday             string,
    d_weekend             string,
    d_following_holiday   string,
    d_first_dom           int,
    d_last_dom            int,
    d_same_day_ly         int,
    d_same_day_lq         int,
    d_current_day          string,
    d_current_week         string,
    d_current_month        string,
    d_current_quarter      string,
    d_current_year          string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/date_dim';

create external table time_dim
(
    t_time_sk          int,
    t_time_id          string,
    t_time               int,
    t_hour                int,
    t_minute              int,
    t_second              int,
    t_am_pm              string,
    t_shift                string,
    t_sub_shift           string,
    t_meal_time           string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/time_dim';

create external table item
(
    i_item_sk          int,
    i_item_id          string,
    i_rec_start_date    timestamp,
```

```

    i_rec_end_date          timestamp,
    i_item_desc             string,
    i_current_price         float,
    i_wholesale_cost        float,
    i_brand_id               int,
    i_brand                  string,
    i_class_id                int,
    i_class                  string,
    i_category_id              int,
    i_category                string,
    i_manufact_id              int,
    i_manufact                string,
    i_size                     string,
    i_formulation              string,
    i_color                     string,
    i_units                     string,
    i_container                string,
    i_manager_id                int,
    i_product_name              string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/item';

create external table store
(
    s_store_sk                 int,
    s_store_id                string,
    s_rec_start_date           timestamp,
    s_rec_end_date              timestamp,
    s_closed_date_sk            int,
    s_store_name                string,
    s_number_employees           int,
    s_floor_space                int,
    s_hours                     string,
    s_manager                    string,
    s_market_id                  int,
    s_geography_class            string,
    s_market_desc                string,
    s_market_manager              string,
    s_division_id                int,
    s_division_name              string,
    s_company_id                  int,
    s_company_name                string,
    s_street_number              string,
    s_street_name                string,
    s_street_type                  string,
    s_suite_number                string,
    s_city                      string,
    s_county                     string,
    s_state                      string,
    s_zip                        string,
    s_country                     string,
    s_gmt_offset                  float,
    s_tax_percentage                float
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/store';

```

```

create external table customer
(
    c_customer_sk          int,
    c_customer_id           string,
    c_current_cdemo_sk     int,
    c_current_hdemo_sk     int,
    c_current_addr_sk      int,
    c_first_shipto_date_sk int,
    c_first_sales_date_sk  int,
    c_salutation           string,
    c_first_name            string,
    c_last_name             string,
    c_preferred_cust_flag   string,
    c_birth_day              int,
    c_birth_month             int,
    c_birth_year              int,
    c_birth_country           string,
    c_login                  string,
    c_email_address           string,
    c_last_review_date        string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/customer';

create external table promotion
(
    p_promo_sk          int,
    p_promo_id           string,
    p_start_date_sk      int,
    p_end_date_sk         int,
    p_item_sk             int,
    p_cost                 float,
    p_response_target     int,
    p_promo_name           string,
    p_channel_dmail        string,
    p_channel_email        string,
    p_channel_catalog       string,
    p_channel_tv             string,
    p_channel_radio           string,
    p_channel_press           string,
    p_channel_event           string,
    p_channel_demo             string,
    p_channel_details          string,
    p_purpose                  string,
    p_discount_active          string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/promotion';

create external table household_demographics
(
    hd_demo_sk          int,
    hd_income_band_sk     int,
    hd_buy_potential       string,
    hd_dep_count             int,
    hd_vehicle_count          int
)
stored as orc

```

```

location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/household_demographics';

create external table customer_address
(
    ca_address_sk          int,
    ca_address_id           string,
    ca_street_number        string,
    ca_street_name          string,
    ca_street_type          string,
    ca_suite_number         string,
    ca_city                 string,
    ca_county               string,
    ca_state                string,
    ca_zip                  string,
    ca_country              string,
    ca_gmt_offset            float,
    ca_location_type         string
)
stored as orc
location 's3://public-qubole/datasets/tpcds/nonpartitioned/orc/scale_1000/customer_address';

```

2.2.16 Performance Monitoring with Ganglia

Ganglia is a scalable, distributed system designed to monitor clusters and grids while minimizing the impact on the performance. When you enable Ganglia monitoring on a cluster, you can view the performance of the cluster as a whole as well as inspect the performance of individual node instances. You can also view various Hadoop metrics for each node instance.

How to Enable Ganglia Monitoring

Perform the following steps to enable Ganglia Monitoring:

1. Sign into the Qubole account.
2. Navigate to **Control Panel** and the **Clusters** tab is displayed by default. Click the edit button of the cluster for which you want to enable Ganglia monitoring. See [Modifying Clusters Configuration](#) for more information.
3. In the **Edit Cluster** page, select **Enable Ganglia Monitoring** that is within **Cluster Settings**. See [Modifying Cluster Settings](#) for more information. The setting is applied when the cluster is restarted as it does not apply to a running cluster.

How to View Ganglia Metrics

Navigate to https://api.qubole.com/ganglia-metrics-<cluster_id> to view the Ganglia metrics of a specific cluster.

Collecting Cluster Metrics

When Ganglia monitoring is enabled on a cluster, you can also collect the cluster metrics using the [Cluster Metrics API](#).

2.3 Qubole Clusters

2.3.1 Introduction to Qubole Clusters

Background

Qubole Data Service (QDS) can run queries and programs written using different interfaces like SQL, Map-Reduce, Cascading, Pig, Scala and Python. These applications are executed using distributed execution frameworks like Hadoop, Spark and Presto. All of these frameworks run on a multi-node compute cluster with one master node and one or more slave nodes. QDS provides a unified platform for managing compute clusters of different types.

The Basics

Each QDS account has:

1. Multiple pre-configured clusters (and any additional ones that the customer might create).
2. Each cluster has a different **Type** corresponding to the execution framework it is configured to run.
3. Each cluster can have one or more unique **Label** (s) associated with it. We will cover *Cluster Labels* in more detail later.
4. New accounts come preconfigured with one each of the following types:
 1. Hadoop-1 (labelled as *default*)
 2. Presto (labelled as *presto*)
 3. Spark (labelled as *spark*)
 4. Hadoop-2 (labelled as *hadoop2*)

The list of all clusters can be seen by clicking on Control Panel in the Left Menu and then clicking on the Clusters tab. The image below shows the preconfigured clusters for a new account.

Active Cluster(s)		Deleted Cluster(s)			
Search : <input type="text" value="Enter Search Text"/>					
Id	Labels	Nodes	Up Time	Resources	Action
4339	default	0			Down ✓ Edit ...
4340	presto	0			Down Edit ...
4341	hadoop2	0			Down Edit ...
4342	spark	0			Down Edit ...

Note:

- The clusters are only configured but not in running state. The red status icon indicates that the cluster is in a *Down* state.
- Users can configure as many clusters of any given Type as they want (Trial accounts may be limited to a smaller number).

Cluster LifeCycle Management

The life cycle of a cluster is best explained by detailing what happens behind the scenes as users and applications interact with QDS

Cluster Bringup

- A cluster is launched automatically when it is needed. For example:
 - if a Hadoop Map-Reduce job is run then a Hadoop-1 cluster may be launched
 - if a Spark application is run then a Spark cluster may be launched
- A cluster can also be launched manually by clicking on the Start button in the Cluster tab above.
- Both the above actions can be performed either from the UI or via API/SDK calls.

Note that many Hive commands (metadata operations like a *show partitions* command) do not require a working cluster. QDS detects this automatically.

Cluster Auto-Scaling

- Clusters are scaled automatically based on the requirements of the workload.
 - QDS continually monitors the current load and adds or removes nodes as required.
- Each of the distributed execution frameworks - Hadoop/Presto/Spark - has been enhanced by Qubole for this purpose
- QDS only removes nodes when they are about to hit the hour boundary.
- AWS bills hourly for machines. So this optimizes the spend in AWS. We adopt a similar strategy in other clouds like GCE to provide a predictable experience
- QDS also removes nodes only after destaging any important data resident on it.
- Important data on a node includes replicas of HDFS blocks and intermediate output from tasks and executors

Cluster Termination

- A cluster is kept running as long as there are active user sessions that have run workload against the cluster.
- A cluster is also kept running as long as there are any jobs or applications running therein.
- If there are no active user sessions or active workloads - QDS may terminate the cluster.
- The exact time of termination is determined by when the nodes are launched. QDS tries to not waste any time on the machines that is already paid for.
- A cluster can also be terminated manually by the user from the Cluster Tab - or via issuing the cluster termination API call.

Cluster Labels and Command Routing

Each cluster in QDS has one or more unique Labels assigned to it. Every account has a Hadoop-1 cluster with the label *default*. Qubole Commands are routed to clusters using some simple rules:

- If a Label is supplied for a command, then the command is routed to the cluster with the corresponding Label.
- If no Label is supplied then the command is routed to the first matching cluster for the given command type. For example:
 - Hive, Pig, Hadoop commands are routed to a cluster of type Hadoop-1
 - Presto commands are routed to a cluster of type Presto

- Spark commands are routed to a cluster of type Spark
- If no Label is supplied - Hive, Pig, Hadoop commands are routed to the Hadoop-1 cluster labelled *default* by default.

Qubole Cluster EC2 Tags

For AWS clusters, Qubole instances are tagged with the following three EC2 tags:

- **Qubole** - This tag's value is a unique identifier based on the account and cluster. Its value is `qbol-acc<AccountID>_cl<ClusterID>`.
- **alias** - This tag identifies the node within the Hadoop cluster. Its value is *master* or *node<number>*.
- **Name** - This tag also identifies the node. Its value is the same as *alias* tag. This value can be overridden using a custom EC2 tag.

For information on custom EC2 tags, see [Modifying EC2 Settings](#) through UI and `hadoop-settings` through an API.

2.3.2 Configuring Clusters

Qubole believes in making Big Data analysis easy. Every new account is therefore configured with some clusters by default and these are sufficient to run small test workloads. This section describes the configuration of Qubole clusters. A good understanding of the tradeoffs and capabilities in this area can help you achieve higher performance and stability at lower costs.

The following section covers these sub-topics:

- [Cluster Settings Page](#)
- [General Cluster Configuration](#)
- [Cluster Type](#)
- [Cluster Size and Instance Types](#)
- [Node Bootstrap File](#)
- [Cluster Composition](#)
- [EC2 Settings](#)
- [Miscellaneous Settings](#)
- [Hadoop-specific Options](#)

Cluster Settings Page

To configure any cluster, go to its Cluster Settings page as follows:

1. Click **Control Panel** on the left navigation pane and click the [Clusters](#) tab.



2. Click the Edit Cluster button on the cluster that you want to modify.

See [Managing Clusters](#) for more information on modifying cluster configuration.

The following sections explain the different options available on the Cluster Settings page.

General Cluster Configuration

Many of the cluster configuration options are common across different types of clusters. Let us cover them first by going over some of the most important categories.

Cluster Labels

As explained in [Cluster Labels](#), each cluster has one or more labels that are used for routing Qubole Commands. In the first form entry, you can assign one or more comma-separated labels to a cluster.

Cluster Type

As introduced in [Qubole Cluster Fundamentals](#), each account has multiple clusters and of a certain type. These types are:

- **Hadoop:**

This is Qubole's tried and tested map-reduce framework running a version of Hadoop API compatible with Apache Hadoop 1.0. Hadoop is suitable for reasonably long-running jobs and batch-processing workloads. You can also use this for Hive, Pig and Map-Reduce applications of all types (including Cascading).

- **Hadoop 2:**

Hadoop-2 clusters run a version of Hadoop API compatible with Apache Hadoop 2.6. Hadoop-2 clusters use Apache YARN cluster manager and are tuned for running Map-Reduce based applications just like the Hadoop clusters.

- **Presto:**

Presto is the fast in-memory query processing engine from Facebook. This is useful for near-real-time query processing for less complex queries that can be mostly performed in-memory.

- **Spark:**

Spark clusters allow you to run applications based on Apache Spark 1.4.0. Spark has a fast in-memory processing engine that is ideally suited for iterative applications like machine learning. Qubole's offering integrates Spark with the YARN cluster manager.

Cluster Size and Instance Types

From a performance standpoint, this is one of the most critical sets of parameters:

- Setting a **Minimum** and **Maximum Slave Count** for a cluster (in addition to one fixed Master node).

Note: All Qubole Cluster types, Hadoop-1, Hadoop-2, Presto or Spark auto-scale up and down automatically within the minimum and maximum range set in this section.

- **Master and Slave Node Type**

The **Slave Node Type** must be selected based on the characteristics of the application. A memory-intensive application would benefit from memory rich nodes (such as *r3* node types in AWS), while a CPU-intensive application would benefit from higher compute power (such as the *c3* node types in AWS).

The **Master Node Type** is usually driven by the size of the cluster. For smaller clusters and workloads, small instances (like *m1.large* in AWS) suffice. But for extremely large clusters (or for running a large number of concurrent applications), Qubole recommends large memory machines (such as *8xlarge* machines in AWS).

Linux/UNIX is used to set up Master and Slave nodes, and run the job. Qubole uses Linux/UNIX instances as cluster nodes. Qubole supports various instance types as master and slave nodes, which are available on the AWS Linux/UNIX platform.

Node Bootstrap file

Advanced applications often require custom software to be installed as a pre-requisite. A mapper Python script may require access to SciPy/NumPy and this is often best arranged by just installing these packages (by using *yum* for example). See [About Node Bootstrap Script](#) for more information.

This field provides the location to a bash script used for installing custom software packages on cluster nodes. The node bootstrap file is executed on both Master and Slave nodes with *root* privileges. On Slave nodes, make sure that the node bootstrap is run before any task is launched on behalf of the application. Note that the storage credentials associated with the account are used for reading the node bootstrap script.

Note that Qubole does not check the exit status of the node bootstrap. If software installation fails and it is unsafe to run user applications on such a machine, you should shut the machine down from the bootstrap script.

See [Running Node Bootstrap and Adhoc Scripts on a Cluster](#) for more information on running node bootstrap scripts.

Cluster Composition

Qubole can launch clusters to optimize the cloud usage costs. This is done by using cheaper instances available at auctioned prices on the AWS Spot market. This section covers different settings relevant to leveraging Spot instances for a specific cluster.

A glossary of terms is useful to understand the settings below:

1. **Core** nodes refer to the Master and initial set of Slave nodes (corresponding to the minimum cluster size)
2. **Autoscaled** nodes refers to nodes that are added beyond the Core nodes
3. **Volatile** Instances are instances that can be lost at any time because they are spot instances obtained at low bid prices. They can be lost when the spot instance price is above the bid rate at which that spot instance was obtained.
4. **Stable** Instances are instances that are unlikely to be lost. They can be:
 - On-demand instances
 - Reserved instances
 - Spot instances with a bid price higher than volatile spot instances

The Cluster Composition section contains the following settings:

- **Autoscaling node purchasing option:** Specifies the AWS purchasing model for slave nodes added to the cluster during automatic scaling. The various values that this option can have are:
 - *On-Demand Instance*:
Qubole recommends this setting for workloads that are not cost sensitive or are extremely time sensitive.
 - *Spot Instance*:
When this option is selected, QDS bids for Spot Instances when provisioning Autoscaled nodes Additional settings under the section *Volatile Spot Instance Settings* become applicable as follows:
 - * **Use Qubole Placement Policy**:
When using spot instances for slaves, this policy ensures that at least one replica of each HDFS block is placed on Stable instances. It is recommended to keep this enabled when using spot instances.

- * **Fallback to on-demand:**

When upscaling the cluster, sometimes Qubole may not be able to procure Spot Instances because of low availability or high price. This option specifies that autoscaling should then fall back to procuring On-Demand instances. This will increase the cost of running the cluster, but ensures that the processing completes relatively quickly. Enable this if command processing time is important to you.

- * **Maximum bid price:**

Specifies the maximum bid price for the Spot nodes as a percentage of On-Demand Instance price.

- * **Request Timeout:**

Specifies the time in minutes for which the bid is valid.

- * **Spot Instances Percentage:**

Specifies the maximum percentage of Spot Instances in the autoscaled nodes. For example, if the cluster has a minimum size of 2 nodes, a maximum size of 10 and a Spot percentage of 50%, then, when scaling the node from 2 to 10 nodes, QDS will try to use 4 On-Demand and 4 Spot nodes.

EC2 Settings

- **Compute Access Key** and **Secret Key** set the AWS credentials used to launch this cluster.

Different clusters can be launched with different pairs of Access/Secret Keys.

- **AWS Region** and **Availability Zone** can be selected for nodes in the cluster.

By default, the Region is set to *us-east-1*. If it is set to *No Preference*, the best Availability Zone based on system health and available capacity is selected. Note that all cluster nodes will be in the same availability zone. You can select an Availability Zone in which instances are reserved to benefit from the AWS Reserved Instances.

- **VPC** and **Subnet** settings can be used to launch this cluster within a specific VPC.

If they are not specified, cluster nodes are placed in a Qubole created Security Group in EC2-Classic or in the *default VPC* for EC2-VPC accounts.

- **EBS Volume** settings allow you to optionally add additional EBS drives to launched instances.

These options are only displayed on instances with limited instance-store capacity (such as *c3* instance types). They are also not displayed for Presto Clusters (which do not need larger local disk storage). Additional drives may not be required if jobs do not generate a lot of output, so use this according to the requirements.

Qubole's software auto-magically uses instance-store drives in preference to EBS volumes even if the latter are configured. This reduces AWS cost and helps to achieve higher and more predictable performance when reading/writing to local disks.

- **Custom EC2 Tags** allow you to tag instances of a cluster to get that tag on AWS. This tag is useful in billing across teams as you get the AWS cost per tag, which helps to calculate the AWS costs of different teams in a company.

Miscellaneous Settings

- **Enable encryption of instance local storage:**

Provides an option to encrypt the data at rest on the node ephemeral (local) storage. If this option is selected, HDFS along with the intermediate output generated by Hadoop is stored in an encrypted form on the underlying storage device. Block device encryption is setup for ephemeral drives before the node joins the cluster. As a side effect, the cluster might take additional time (depending on the instance type selected) before it becomes operational.

- **Disable Automatic Cluster Termination:**

By default, an idle cluster is automatically terminated. This option allows you to override this behavior. If the option is enabled, the cluster can still be down-scaled to minimum slave count but it will not be terminated automatically. It must be terminated explicitly by you. Use this option with extreme caution.

- **Enable Ganglia Monitoring:**

This provides an option to monitor the state of cluster with Ganglia. When you enable Ganglia monitoring for a cluster, you can view the performance of the cluster as a whole, and as well inspect the performance of individual node instances. You can also view various CPU and disk metrics as well as detailed Hadoop metrics. For more information on Performance monitoring with Ganglia, refer to [Performance Monitoring with Ganglia](#).

Hadoop-specific Options

- **Override hadoop configuration variables:** Provides an option to override the default Hadoop cluster and job configurations. Provides one key value pair per line.
- **Recommended configuration:** The recommended configuration variable values, that are used by default, are displayed in this section for reference.
- **Fair scheduler configuration:** Provides the fair scheduler configuration values.
- **Default fair scheduler pool:** All jobs on the cluster default to this pool, unless overridden at job submission time.

2.3.3 Cluster Operations

The following section explains the cluster operations such as start, terminate, add, clone, and delete a cluster.

How to Start or Terminate a Cluster

The current state of the cluster is indicated by the icon on the extreme left of each row in the cluster table. It is a green circle for a running cluster, a red circle for a terminated cluster and an arrow pointing up or down for clusters that are starting up or terminating, respectively.



- Click the start icon to start a cluster or restart a stopped cluster.

A dialog box is displayed prompting for confirmation. Click **Ok** to start a cluster. Click **Cancel** if you do not want to start a cluster.



- Click the stop icon to terminate a running cluster.

A dialog box is displayed prompting for confirmation. Click **Ok** to terminate a cluster. Click **Cancel** if you do not want to terminate a cluster.



Click the refresh icon that is on the top-right corner of the **Clusters** page to refresh the clusters status.

How to Add a Cluster



To add a cluster, click the add icon that is on the top-right corner of the **Clusters** page.

The **Create New Cluster** page is displayed. Enter the following details to create a new cluster:

- **Cluster Labels:** A list of comma-separated labels to uniquely identify the cluster. This is a mandatory field.
- For more information on the rest of the parameters, refer to [Configuring Clusters](#) and [Modifying Cluster Configuration](#) for more information.

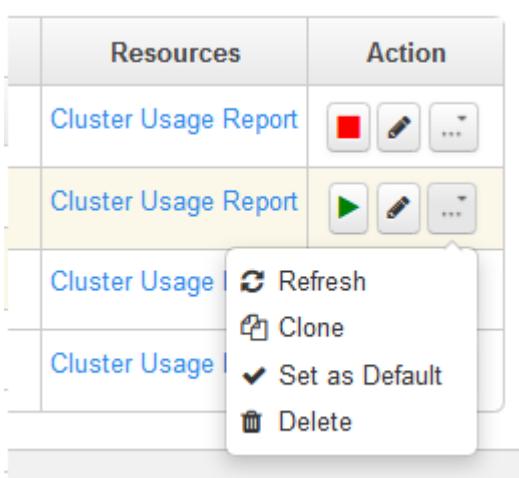
Click **Save** to create the new cluster.

How to Clone a Cluster

Cloning may be preferable to creating a new cluster in many cases since most of the fields will be copied from an existing cluster.

To clone a cluster, click the ellipse icon  listed in the **Action** column.

Select **Clone** from the list of options as shown in the following figure.



The **Clone a Cluster** page is displayed. Enter a new label for the cluster. Do the required modifications and click **Save** to clone the cluster. The label is the only mandatory field to be changed when you clone a cluster.

Clicking **Clone** takes you to the **Edit Clusters** page.

How to Modify a Cluster

To edit a cluster, click the edit icon  available in the **Action** column.

The **Edit Cluster** page is displayed. The current configuration of the cluster is displayed on this page. You can make the desired modifications to it. See [Modifying Cluster Configuration](#) for more information. Click **Save** to save the modifications.

How to Push Configuration to a Cluster

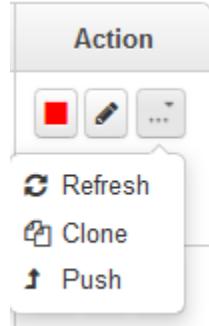
While most cluster attributes only take effect only when a cluster is restarted, some can be *pushed* to a running cluster. The following cluster attributes can be pushed to a running cluster:

- The maximum size of the cluster

- The fair scheduler configuration
- The maximum spot bid price for slave instances

To push configuration to a running cluster, click the ellipse  icon listed in the **Action** column.

Select **Push** from the list of options as shown in the following figure.



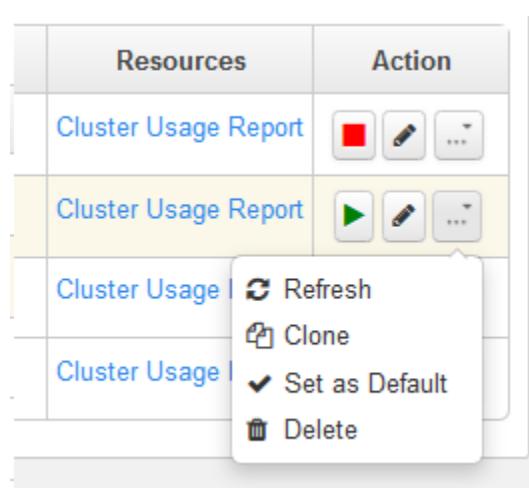
The **Edit Cluster** page with the existing configuration is displayed but with only the *pushable* fields as editable.

Do the changes and click **Push** to make them to be effective immediately.

How to Delete a Cluster

To delete a cluster, click the ellipse  icon listed in the **Action** column.

Select **Delete** from the list of options as shown in the following figure.



A dialog box is displayed prompting for confirmation. Click **Ok** to delete a cluster. Click **Cancel** if you do not want to delete a cluster.

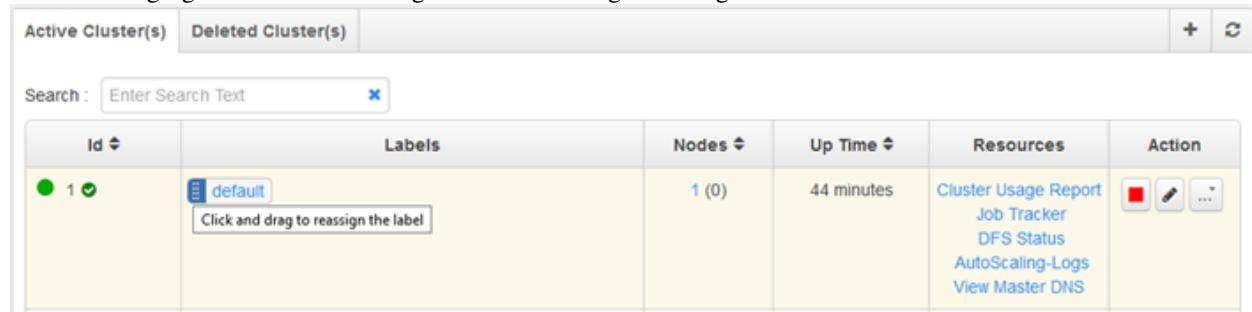
Deleted clusters are visible in the **Deleted Clusters** tab on the clusters page. Once a cluster is deleted, it cannot be retrieved. So, be cautious when deleting a cluster.

Note: Running clusters and the cluster labeled *default* cannot be deleted.

Switching Clusters

Clusters are identified using labels. When you want to switch the workload of one cluster to another, you can reassign the clusters' labels. To do this, click the blue handle  at the left of a label and drag it to another row in the table.

The following figure shows the message to click and drag to reassign the cluster label.



Active Cluster(s)		Deleted Cluster(s)				
Search : Enter Search Text						
Id	Labels	Nodes	Up Time	Resources	Action	
1	default Click and drag to reassign the label	1 (0)	44 minutes	Cluster Usage Report Job Tracker DFS Status AutoScaling-Logs View Master DNS	 	

2.3.4 About Node Bootstrap Script

Qubole allows bootstrapping the nodes in a cluster with custom scripts during the startup. Bootstrap scripts allow installation, management, and configuration of tools useful for cluster monitoring and/or data loading processes. When you specify a node bootstrap script, the node bootstrap script is executed on all cluster nodes including auto-scaled slave nodes when they come up.

The scripts must be placed in the default location `scripts/` directory. (for example: `s3://pydata.com/scripts/node_bootstrap.sh`). The logs of the node bootstrap file are in `node_bootstrap.log` placed in `/media/ephemeral0/logs/others`.

Node bootstrap script is called as part of the user-data-file execution in an AWS cluster. It is invoked as a root user. It does not have a terminal (TTY or text-only console) and many programs do not run without a TTY. In Hadoop clusters, a node bootstrap is invoked after HDFS daemons have been brought up (Namenode on master and DataNode on slaves) but before Map-Reduce/YARN daemons have been initialized (JobTracker/TaskTrackers) that is Hadoop applications are run only after the node bootstrap has been run.

Node bootstrap is idempotent and it is executed via data attached to the instance. The data on the instance is executed only on the first boot cycle but it is not executed on reboot. So, node bootstrap is also executed on the first boot cycle.

Cluster bringup waits for the node bootstrap execution. If the bootstrap takes too long to execute, the cluster bringup can timeout and fail. If the bootstrap takes long time, then cluster bringup does not fail. There is no limit on the time period for node bootstrap execution. Only worker daemons/task execution daemons, for example, NodeManager (Hadoop2) and TaskTracker (Hadoop1) wait for the bootstrap execution.

[Running Node Bootstrap Scripts on a Cluster](#) describes how to run node bootstraps on a cluster and [Run Utility Commands in a Cluster](#) describes how to run utility commands to get the node-related information such as seeing if a node is slave/master and master IP address.

2.3.5 Running Node Bootstrap and Adhoc Scripts on a Cluster

Qubole allows running node bootstrap scripts and scripts in an adhoc manner as required on a cluster. The following topics explain about running node bootstrap and adhoc scripts:

- [Running Node Bootstrap Scripts on a Cluster](#)
- [Examples of Bootstrapping Cluster Nodes with Custom Scripts](#)

- *Running Adhoc Scripts on a Cluster*
- *Limitations of Running Adhoc Scripts*

Running Node Bootstrap Scripts on a Cluster

Qubole allows bootstrapping the nodes in a cluster with custom scripts during the startup. Bootstrap scripts allow installation, management, and configuration of tools useful for cluster monitoring and/or data loading processes. When you specify a node bootstrap script, the node bootstrap script is executed on all cluster nodes including auto-scaled slave nodes when they come up.

The scripts must be placed in the default location scripts/ directory. (for example: s3://pydata.com/scripts/node_bootstrap.sh). You can edit the default script used for node_bootstrapping in the **Cluster Settings** configuration. (Navigate to **Cluster Settings** through **Control Panel > Clusters > Edit Cluster**). See [Modifying Cluster Settings](#) for more information.

The logs of the node bootstrap file are in *node_bootstrap.log* placed in */media/ephemeral0/logs/others*. See [About Node Bootstrap Script](#) for more information.

The following illustration shows an example of **Cluster Settings** of a cluster where the node bootstrap file can be edited in the **Node Bootstrap File** text box (at the end of the default bucket location).

[Cluster > Edit Cluster\(2166\)](#)

The screenshot shows the 'Cluster Information' tab selected in the navigation bar. Below it, the 'Cluster Settings' section is displayed. The 'Node Bootstrap File' field contains the value 's3://pydata.com/scripts/ node_bootstrap.sh'. Other settings shown include Cluster Labels ('CustomerJobs'), Cluster Type ('Hadoop'), and various slave/master node type configurations.

Examples of Bootstrapping Cluster Nodes with Custom Scripts

Example 1: Using Giraph Giraph is a framework to perform offline batch processing of semi-structured graph data on a massive scale. To use Giraph jar files in a bootstrap script, perform the following steps:

1. Use Giraph jar files and create a node bootstrap script, node_bootstrap.sh.

```
mkdir -p /media/ephemeral0/
```

```
cd /media/ephemeral0/
hadoop dfs -get s3://paid-qubole/giraph/giraph-qubole.tar.gz .
tar -xvf giraph-qubole.tar.gz
```

2. Run the Hadoop Jar Jobs

Sample Shortest Path Job

```
hadoop dfs -put /media/ephemeral0/giraph/tiny_text.txt /tmp
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.45.x86_64
hadoop jar /media/ephemeral0/giraph/giraph-examples/target/giraph-examples-1.1.0-SNAPSHOT-for-hadoop-
```

Page Ranker Benchmark Job

```
hadoop jar /media/ephemeral0/giraph/giraph-examples/target/giraph-examples-1.1.0-SNAPSHOT-for-hadoop-
```

Example 2: Changing Python Version On cluster nodes, the default version of Python is 2.6. It is possible to use Python 2.7 for Hadoop tasks by adding the following lines to the node bootstrap file specified in the cluster configuration.

```
source /usr/lib/hustler/bin/qubole-bash-lib.sh
qubole-hadoop-use-python2.7
```

Example 3: Node Bootstrap Script for a Mahout Job

```
mkdir -p /media/ephemeral0/mahout
cd /media/ephemeral0/mahout
hadoop dfs -get s3://paid-qubole/mahout0.9/mahout-distribution-0.9.tar.gz .
hadoop dfs -get s3://paid-qubole/mahout0.9/data.tar.gz .

#copy stuff from s3://paid-qubole

tar -xvf mahout-distribution-0.9.tar.gz
tar -xvf data.tar.gz
```

Example 4: Adding Node Bootstrap Script on AWS EC2 Cluster using Pig

1. Create a file named node_bootstrap.sh (file name is user-defined) with the content:

```
mkdir -p /media/ephemeral1/pig11
hadoop dfs -get s3://paid-qubole/pig11/pig.tar.gz /media/ephemeral1/pig11/
tar -xvf /media/ephemeral1/pig11/pig.tar.gz -C /media/ephemeral1/pig11/
```

2. Edit the specific Qubole cluster in the **Control Panel** and enter the name of the Bootstrap file into the location at **Node Bootstrap File** and place the file in the appropriate location in AWS S3.

With the above example of Bootstrap file, pig11 gets installed on all the nodes of cluster. Later, you can use a *Shell Command* interface from the QDS user interface and invoke any pig script using the pig command. You must provide the complete path of pig as shown in the following example.

```
/media/ephemeral1/pig11/pig/bin/pig s3n://<BucketName>/<subfolder>/<PigfileName>.pig
```

Example 5: Installing R and RHadoop on cluster

1. Create a file named node_bootstrap.sh (file name is user-defined) with the content:

```
sudo yum -y install R
echo "install.packages(c(\"rJava\", \"Rcpp\", \"RJSONIO\", \"bitops\", \"digest\",
  \"functional\", \"stringr\", \"plyr\", \"reshape2\", \"dplyr\",
  \"R.methodsS3\", \"caTools\", \"Hmisc\"), repos=\"http://cran.uk.r-project.org\")" > Rscript base.R
wget https://github.com/RevolutionAnalytics/rhdfs/blob/master/build/rhdfs_1.0.8.tar.gz
echo "install.packages(\"rhdfs_1.0.8.tar.gz\", repos=NULL, type=\"source\")" > rhdfs.R
Rscript rhdfs.R
wget https://github.com/RevolutionAnalytics/rmr2/releases/download/3.3.1/rmr2_3.3.1.tar.gz
echo "install.packages(\"rmr2_3.3.1.tar.gz\", repos=NULL, type=\"source\")" > rmr.R
Rscript rmr.R
cd /usr/lib/hadoop
wget http://www.java2s.com/Code/JarDownload/hadoop-streaming/hadoop-streaming-1.1.2.jar.zip
unzip hadoop-streaming-1.1.2.jar.zip
```

2. Edit the specific Qubole cluster in the **Control Panel** and enter the name of the Bootstrap file into the location at **Node Bootstrap File** and place the file in the appropriate location in AWS S3.

With the above example of Bootstrap file, R, RHadoop and RHDFS get installed on the cluster. One can run R commands as well as RHadoop commands. A sample R script using RHadoop is as given below.

```
Sys.setenv("HADOOP_STREAMING"="/usr/lib/hadoop/hadoop-streaming-1.1.2.jar")
library(rmr2)
small.ints = to.dfs(1:1000)
mapreduce(
  input = small.ints,
  map = function(k, v) cbind(v, v^2))
```

Running Adhoc Scripts on a Cluster

At times, you want to execute some scripts on the cluster in an adhoc manner. To run an adhoc script, you can use a REST API to execute a script located in S3 on the cluster. See *Run Adhoc Scripts on a Cluster* for the REST API information.

The Run-Adhoc Script functionality uses the pssh functionality to spawn adhoc scripts on the cluster nodes and it has been tested under the following conditions:

- Works in clusters that are being set up using a proxy tunnel server
- Even if the script execution time is longer than the pssh timeout, the script still executes on the node.

Limitations of Running Adhoc Scripts

If a script is already being executed and you try to execute the same script, it does not execute as a current instance of the script is already being executed. To avoid this, you can tweak the path of S3 script, which can be executed as a separate instance of this API on the cluster.

2.3.6 Cluster Network Security Characteristics

Each cluster is associated with a unique security group which acts as a virtual firewall that controls the traffic for the nodes within the cluster. The ports, which allow inbound traffic are as listed below:

- Ports that allow inbound traffic from the Qubole's security group sg-a8c407c0 in the us-east-1 region are:
 - Port 22, which is the SSH port

- Port 9000, which is the NameNode port
- Port 9001, which is the JobTracker port
- Port 50030, which is the JobTracker web port
- Port 50070, which is the NameNode web port
- Port 50060, which is the TaskTracker web port
- Port 50075, which is the DataNode web port
- Port 8081, which is the Presto server port
- Port 22 allows inbound traffic from:
 - Qubole's security group `sg-a8c407c0` in the `us-east-1` region and the AWS EC2-classic platform.
 - CIDR `0.0.0.0/0` (world) for all other cases, which includes clusters in VPC (including default VPC in the `us-east-1` region) and regions other than `us-east-1`.

Note: Send an email to help@qubole.com if you want to restrict SSH port (port 22) access to limited IP addresses.

- Within a cluster, participating nodes can communicate with each other on all ports.

2.4 Hive in Qubole

2.4.1 Differences from Apache

Qubole's hive distribution is derived from the Apache Hive version 0.13. However, there are a few differences in the functionality.

Map Joins

The Automatic conversion of the map-join feature is disabled in Qubole's hive. However, if you wish to avail optimization of the feature, do the following.

```
set hive.auto.convert.join=true;
```

NOTE: It is recommended to use a conditional task. However, if the map-join fails for any reason, the following setting will ensure a sort-merge based join.

```
set hive.auto.convert.join.noconditionaltask=false;
```

S3 Listing Optimization

As part of the split computation, the Hive needs to list all the files in the table's S3 location. The implementation in Apache Hadoop for listing files in S3 is very slow. Optimizations have been incorporated to speed this up. The following parameter is enabled by default.

```
set fs.s3.inputpathprocessor=true;
```

2.4.2 Data Model

Data in QDS are organized as tables and table partitions. These tables and partitions can either be created from data that the user already has in a S3 bucket or can be generated as an output of running Hive queries. QDS uses [HiveQL](#) to access and query these tables and partitions. For a primer on Hive, refer to the [Apache Hive wiki](#).

The following topics are described in this section:

- [Types of Hive Tables](#)
- [Usage Scenarios](#)
- [External tables](#)
- [Regular Tables](#)
- [Temporary Tables](#)
- [Configuring Table Storage](#)
- [Default Tables](#)
- [Partitions](#)

Types of Hive Tables

Tables in QDS are backed by data residing in S3. However, they can also be backed by data stored in HDFS (Hadoop Distributed File System). To understand the different storage choices, it is required to first understand the basic table types available in (the QDS version of) Hive:

- **External Tables:** These tables are assigned an explicit location by the user. When an external table is dropped, Hive does not delete the data in the location that it points to.
- **Regular Tables:** These tables do not have an explicit location attribute and are assigned by one by Hive directly. Hive assigns a location relative to a default location (that is fixed per account). When a regular table is dropped, the data in the table is deleted.
- **Temporary Tables:** The QDS version of Hive allows a third form of tables that is deleted automatically once the user's session is deleted.
- **Mongo backed Tables:** This allows users to create a Hive table in which the underlying data sits in a Mongo DB collection. Qubole dynamically queries the Mongo DB instance to fetch data when the table is queried. See [Mongo Backed Tables](#) for more details.

The notion of default location is explained in the section on Configuring Table Storage.

Usage Scenarios

The rationale for the above classes of table storage can be explained through some simple examples applicable to QDS. In order to go through these examples, use the example dataset derived from the [Wikipedia Page Traffic Statistics](#) dataset that is uploaded at paid-qubole AWS bucket (paid-qubole/default-datasets/miniwikistats/).

The mini-wikistats dataset differs from the original in being reduced in size. It only has data from the last hour of each day and only has data for 7 days. This bucket is a public (requester-pays) bucket. Note that the full data is available at [Wikipedia Page Traffic Statistics](#).

The examples below will work against the original and larger dataset simply by removing the prefix **mini** in the examples mentioned below.

External Tables

A typical user will have pre-existing data in some S3 bucket. An external table can be created over such data to begin analyzing it. For instance:

```
CREATE EXTERNAL TABLE
miniwikistats (projcode string, pagename string, pageviews int, bytes int)
PARTITIONED BY (dt string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '
LINES TERMINATED BY '\n'
LOCATION 's3n://paid-qubole/default-datasets/miniwikistats/';
```

This table is used as a reference in subsequent examples. Hence, it is worth diving into the assumptions built into this statement.

1. The data is assumed to be entirely in text and contains 4 fields that are separated using space character and the rows are separated by newline character.
2. The data lives under the location s3n://paid-qubole/default-datasets/miniwikistats/.
3. Finally, the dataset is assumed to be partitioned (by hour for example) so that each hour's statistics is in a separate subfolder within that location. For example, the statistics for first hour of 20110101 is in the location s3n://paid-qubole/default-datasets/miniwikistats/20110101-01/.
4. Note that the AWS secret and access keys can be provided as part of the table location url, for example:

```
s3n://ACCESS_KEY:SECRET_KEY@paid-qubole/default-datasets/miniwikistats/20110101-01/
```

Note: The statement above creates a partitioned table. However, it does not populate any partitions in it. Without populating partitions, the table is empty (even though this S3 location has data). To populate partitions in this table, refer to the section on Partitions. Run the first command mentioned in the Partitions section and continue to the examples mentioned below.

Regular Tables

The user may create derived data sets while analyzing the original data sets and may want to keep them for a long lived period. In such a case, a regular table can be created, which will also be located in S3, for instance:

```
CREATE TABLE q1_miniwikistats
AS
SELECT projcode, sum(pageviews)
FROM miniwikistats
WHERE dt >= '20110101' AND dt <= '20110105'
GROUP BY projcode;
```

Temporary Tables

The user may want to force a table to be located in HDFS to take advantage of its higher speed and/or the fact that it is automatically deleted at the end of the session:

```
CREATE TMP TABLE tmp_stats AS
SELECT * FROM miniwikistats
WHERE projcode LIKE 'ab%' AND dt > '20110101';
```

The location of all tables created in this manner can be looked up using the describe command:

```
DESCRIBE FORMATTED tmp_stats;
```

Configuring Table Storage

Storage for temporary tables in QDS Hive is always in HDFS/Hadoop. Similarly, storage for external tables is relatively simple to understand. The user can explicitly list the entire location. So the user has to only worry about configuring the S3 storage for regular tables.

As part of setting up an account, users can set a default location in S3, and credentials used to access (read/write/list) this location. QDS creates regular tables in a subdirectory of this location. (Other subdirectories under the same folder are used to store logs and results). Locations for external tables can specify credentials as part of the location URL. However, this is only required if external table locations are not accessible using the storage credentials specified in the account.

Default Tables

To help users get started, for each account, Qubole creates some *read-only* tables, by default. These tables are backed by a public *requester-pays* bucket in Amazon. The tables created right now are as follows:

- **default_qubole_demo_airline_origin_destination**: This has a subset of the data described [here](#).
- **default_qubole_memetracker**: 96 million memes collected between 2008 and 2009 as described [here](#).

Partitions

Large data sets are typically divided into directories. Directories map to partitions in Hive. Currently, partitions in Hive must be populated manually using the following command (picking up from the previous *create external table* example):

```
ALTER TABLE miniwikistats RECOVER PARTITIONS;
```

Following this command, all the data in the location `s3n://paid-qubole/default-datasets/miniwikistats/` can be queried through the table *miniwikistats* via the [Analyze](#) page.

Note: The partitions are created only for the subdirectories that exist while running this command. If more directories are added later, for example if the data is being continuously loaded, this command must be run again. To see the partitions of that table, execute the following statement:

```
SHOW PARTITIONS miniwikistats;
```

Qubole's version of 'recover partitions' also supports recovering specific partitions using patterns. For example, consider the directory structure of the miniwikistats dataset.

```
/default-datasets/miniwikistats/20110101-01
/default-datasets/miniwikistats/20110101-02
/default-datasets/miniwikistats/20110102-01
/default-datasets/miniwikistats/20110102-02
/default-datasets/miniwikistats/20110103-01
/default-datasets/miniwikistats/20110103-02
/default-datasets/miniwikistats/20110104-01
/default-datasets/miniwikistats/20110104-02
/default-datasets/miniwikistats/20110105-01
/default-datasets/miniwikistats/20110105-02
```

A table pointing to the miniwikistats dataset directory

```
CREATE EXTERNAL TABLE slice_miniwikistats ...
LOCATION 's3n://paid-qubole/default-datasets/miniwikistats/'
```

Then

```
ALTER TABLE slice_miniwikistats RECOVER PARTITIONS LIKE '20110102*' 
```

Altering table recovers two partitions only (20110102-01 and 20110102-02).

2.4.3 Hive Connectors

Big Query - Hive Connector

We can now read from and write to Big query table using Qubole Hive.

There are two ways to connect to BigQuery table using Hive: Read Mode (or) Read and Write Mode.

Add Required Jars

```
add jar gs://qubole-gce-dev/big-query/qubole-hive-connectors-final-0.0.2.jar
```

Read Mode

Hive Table Creation

Table creation is a step. During this, to enable hive to read data, all the BigQuery table properties are passed on. The table creation requires gcs-bucket, project ID, dataset ID, BigQuery table ID properties. The following sections explain how such properties are passed on.

Example

```
drop table if exists big_hive;
CREATE EXTERNAL TABLE big_hive( word string,
                                word_count bigint,
                                corpus string,
                                corpus_date bigint)
STORED BY 'com.qubole.hive.bigquery.HiveBigQueryStorageHandler'
TBLPROPERTIES ( "mapred.bq.gcs.bucket" = "qubole-gce-dev",
                 "mapred.bq.input.table.id" = "shakespeare",
                 "mapred.bq.input.project.id"= "qubole-gce",
                 "mapred.bq.input.dataset.id"= "testing_dataset",
                 "mapred.bq.query.results.table.delete"="true");
```

Query Data

Example

```
select * from big_hive limit 10;
select * from big_hive where word = "answer" limit 10;
```

Read and Write Mode

There are two types of modes: **Read** mode and **Write** mode. Both these modes require different set(s) of parameters. However to enable the write capabilities, a few extra properties are required. For example, to provide the schema of the BigQuery table, the table properties should include "mapred.bq.output.table.schema".

Hive Table Creation

```
drop table if exists big_hive_output;
CREATE EXTERNAL TABLE big_hive_output( word string,
                                         word_count bigint,
                                         corpus string,
                                         corpus_date bigint)
STORED BY 'com.qubole.hive.bigquery.HiveBigQueryStorageHandler'
TBLPROPERTIES ( "mapred.bq.gcs.bucket" = "qubole-gce-dev",
                  "mapred.bq.input.table.id" = "shakespeare_output",
                  "mapred.bq.input.project.id" = "qubole-gce",
                  "mapred.bq.input.dataset.id" = "testing_dataset",
                  "mapred.bq.output.table.schema" =
                  "[
                      {'name': 'word', 'type': 'string'},
                      {'name': 'word_count', 'type': 'integer'},
                      {'name': 'corpus', 'type': 'string'},
                      {'name': 'corpus_date', 'type': 'integer'}
                  ]"
);
```

Note: Write functionality requires an extra dataset to be created, if it does not already exist. The dataset, in which the table exists, is appended with _hadoop_temporary. For example, testing_dataset_hadoop_temporary.

Writing Data

```
insert overwrite table big_hive_output select * from big_hive limit 2;
```

Note: Writing to the BigQuery table is an append only functionality. This does not overwrite any of the existing data.

Query Data

Example

```
select * from big_hive_output limit 10;
select * from big_hive_output where word = "answer" limit 10;
```

DynamoDB - Hive Connector

Now, we can read from and write to a dynamoDB table using Qubole Hive.

Following illustrates the process of connecting a dynamoDB table with a Hive table.

Add Required Jars

```
add jar s3://paid-qubole/dynamoDB/jars/qubole-hive-connectors-final-0.0.2.jar;
```

Set credentials

```
set mapreduce.dynamodb.access.key=FSDFDSFDS;
set mapreduce.dynamodb.secret.key=dfgds jkhfdhfdfsbfdsk;
```

Hive Table Creation

Table creation is the step, where, all the dynamoDB table properties are passed on, enabling hive to access the data. The table creation requires “dynamodb.table.name” and “dynamodb.column.mapping”. These properties are passed on as follows:

Example

```
drop table dynamoDBTest;
CREATE EXTERNAL TABLE dynamoDBTest (
    string_eg string,
    number_eg bigint,
    binary_eg binary,
    strings_eg array<string>,
    numbers_eg array<bigint>,
    binarys_eg array<binary>
)
STORED BY 'com.willetinc.hive.mapreduce.dynamodb.HiveDynamoDBStorageHandler'
TBLPROPERTIES ("dynamodb.table.name" = "dynamoDBTestTable_donot_modify",
                "dynamodb.column.mapping" = "string_eg:s_eg, number_eg:n_eg,
                                            binary_eg:b_eg,
                                            strings_eg:ss_eg,
                                            numbers_eg:ns_eg, binarys_eg:bs_eg"
);

```

Query Data Example

```
select * from dynamoDBTest;
select * from dynamoDBTest where 1==1 limit 10;

insert overwrite table dynamoDBTest select * from hive_table limit 2;
```

Throughput In order to achieve the optimal performance, you can tweak the following parameters:

Parameter	Description
dy-namodb.throughput.readpercentage	Set the rate of read operations to keep your DynamoDB provisioned throughput rate in the percentage for your table. The value is between 0.1 and 1.5, inclusively. Default value is set to 0.5
dy-namodb.max.map.tasks	Specify the maximum number of map tasks when reading data from DynamoDB. This value must be equal to or greater than 1.

HBase - Hive Connector

We can now read from and write to HBase tables using Qubole Hive. HBase version 0.94 is the only supported version.

Add Required Jars

```
add jar s3://paid-qubole/hive11_hbase94/hive-hbase-handler-0.11.0.jar;
add jar s3://paid-qubole/hive08_hbase94/protobuf-java-2.4.1.jar;
add jar s3://paid-qubole/hive08_hbase94/hbase-0.94.7.jar;
add jar s3://paid-qubole/hive08_hbase94/zookeeper-3.3.1.jar;
```

Specify Zookeeper IP Address

```
set hbase.zookeeper.quorum=<IP Address>;
```

Hive tables with data in HBase

HBase tables can be used to store data in a Hive table. These HBase tables are managed by Hive. If the Hive table is dropped, the HBase table is also deleted.

Example

The example below creates a Hive table and the data is stored in a HBase table called ‘pagecounts’

```
CREATE TABLE IF NOT EXISTS pagecounts_hbase (rowkey STRING, pageviews STRING, bytes STRING)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,f:c1,f:c2')
TBLPROPERTIES ('hbase.table.name' = 'pagecounts');
```

Access HBase tables in Hive

Many HBase tables are created and managed independently. Hive can access these tables without managing by a creating an EXTERNAL table.

Example

The example below creates a Hive table and accesses the data is stored in a HBase table called ‘pagecounts’

```
CREATE EXTERNAL TABLE IF NOT EXISTS pagecounts_hbase (rowkey STRING, pageviews STRING, bytes STRING)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('hbase.columns.mapping' = ':key,f:c1,f:c2')
TBLPROPERTIES ('hbase.table.name' = 'pagecounts');
```

Hive-JDBC Connector

Qubole provides a Hive connector for JDBC to read from and write to JDBC databases using Hive and enables them to run SQL queries to analyze data that resides in JDBC tables. Optimizations such as PredicatePushDown have also been added.

The sample queries and POM file are provided in [Hive JDBC Storage Handler](#)

Add Required Jars

```
add jar s3://paid-qubole/jars/jdbchandler/mysql-connector-java-5.1.34-bin.jar;
add jar s3://paid-qubole/jars/jdbchandler/qubole-hive-JDBC-0.0.6.jar;
```

Table Creation

An external Hive table connecting a JDBC table can be created as follows. This enables us to read and write to underlying JDBC table with ease.

Example

Table can be created in two ways:

- First is, Column mappings can be explicitly given along with the table creation statement.

```
DROP TABLE HiveTable;
CREATE EXTERNAL TABLE HiveTable(
    id INT, id_double DOUBLE, names STRING, test INT
)
STORED BY 'org.apache.hadoop.hive.jdbc.storageHandler.JdbcStorageHandler'
TBLPROPERTIES (
    "mapred.jdbc.driver.class"="com.mysql.jdbc.Driver",
    "mapred.jdbc.url"="jdbc:mysql://localhost:3306/rstore",
    "mapred.jdbc.username"="----",
    "mapred.jdbc.input.table.name"="JDBCTable",
    "mapred.jdbc.output.table.name"="JDBCTable",
    "mapred.jdbc.password"="----"
);
```

- Second is, no table mappings are specified, SerDe class automatically generates those mappings.

```
CREATE EXTERNAL TABLE HiveTable
row format serde 'org.apache.hadoop.hive.jdbc.storagehandler.JdbcSerDe'
STORED BY 'org.apache.hadoop.hive.jdbc.storagehandler.JdbcStorageHandler'
TBLPROPERTIES (
    "mapred.jdbc.driver.class"="com.mysql.jdbc.Driver",
    "mapred.jdbc.url"="jdbc:mysql://localhost:3306/rstore",
    "mapred.jdbc.username"="root",
    "mapred.jdbc.input.table.name"="JDBCTable",
    "mapred.jdbc.output.table.name" = "JDBCTable",
    "mapred.jdbc.password"=""
);
```

Usage

Hive-JDBC connector supports almost all types of possible SQL queries. Some examples of supported queries are:

Reading Data

```
> select * from HiveTable;
> select count(*) from HiveTable;
> select id from HiveTable where id > 50000;
> select names from HiveTable;
> select * from HiveTable where names like 'D%';
> select * FROM HiveTable ORDER BY id DESC;
```

Joining Tables

```
> select HiveTable_1.* , HiveTable_2.* from HiveTable_1 a join HiveTable_2 b
on (a.id = b.id) where a.id > 90000 and b.id > 97000;
```

Writing Data

```
> Insert Into Table HiveTable_1 select * from HiveTable_2;
> Insert Into Table HiveTable_1 select * from HiveTable_2 where id > 50;
```

Group By Queries

```
> select id, sum(id_double) as sum_double from HiveTable group by id;
```

Support for PredicatePushDown

To enable/disable PredicatePushDown, add the following configuration.

```
set hive.optimize.ppd = true/false
```

Mongo Backed Tables

QDS allows users to create Hive tables that point to a MongoDB collection. When this table is queried, a MR job is launched that fetches the data from the collection and then further processing is done by Hive. Qubole's implementation of the connector is based on the code at <https://github.com/mongodb/mongo-hadoop>.

Consider the following example SQL statement which points a Hive table to the a MongoDB collection.

```
CREATE EXTERNAL TABLE mongotest (city string, pop int, state string)
STORED BY "com.mongodb.hadoop.hive.MongoHiveStorageHandler"
WITH serdeproperties ("qbol.mongo.input.format"="true")
tblproperties("mongo.input.uri" = "mongodb://<userid>:<password>@<hostname>.mongolab.com:43207/test");
```

This points the table mongotest to a Mongo collection, zips, stored in the test DB in a specific Mongolab hosted instance. Once this table is created, users can then query it like an ordinary Hive table.

To get mongo tables working, add the following setting:

```
set mongo.input.split.create_input_splits=false
```

For example, to compute the population by state, add the following setting:

```
select state, sum(pop) as pop from mongotest group by state
```

You can also use this to extract data out of Mongo and store it in S3 as a normal Hive table.

To limit load on the Mongo database we limit the number of mappers that can connect to each database. By default, this number is set to 4. That is, at the most 4 simultaneous connections are made per map reduce (MR) job to the Mongo DB. In order to change this, add the following setting:

```
set mongo.mapper.count=X;
```

Similarly, for connecting the MR jobs to the read replicas instead of the master, add the following setting:

```
set mongo.input.split.allow_read_from_secondaries=true;
```

When set to true, the MR job tries and uses read replicas whenever possible.

This is just the first step in Qubole's efforts to better the integration of QDS with other data platforms. Qubole expects to be making a number of changes in this area.

2.4.4 Delimited Tables

Data

```
5|ETHIOPIA|0|ven packages wake quickly. regul|
6|FRANCE|3|refully final requests. regular, ironi|
7|GERMANY|3|l platelets. regular accounts x-ray: unusual, regular acco|
8|INDIA|2|ss excuses cajole slyly across the packages. deposits print aroun|
```

DDL

```
create external table nation_s3_txt
(N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE
LOCATION 's3://qtest-qubole-com/datasets/presto/functional/nation_s3_txt';
```

Features

- Entire record must be one line in text file
- Field delimiter configurable via DDL
- Delimited files may be compressed using gzip, bzip2 or lzo

2.4.5 JSON Tables

JSON Serde is useful to parse data stored as JSON. The JSON implementation has been borrowed from the [rcongiu](#).

Data

```
{"n_nationkey": "5", "n_name": "ETHIOPIA", "n_regionkey": "0", "n_comment": "ven packages wake quickly. regul|"}, {"n_nationkey": "6", "n_name": "FRANCE", "n_regionkey": "3", "n_comment": "refully final requests. regular, ironi|"}, {"n_nationkey": "7", "n_name": "GERMANY", "n_regionkey": "3", "n_comment": "l platelets. regular accounts x-ray: unusual, regular acco|"}, {"n_nationkey": "8", "n_name": "INDIA", "n_regionkey": "2", "n_comment": "ss excuses cajole slyly across the packages. deposits print aroun|"}]
```

DDL

```
CREATE EXTERNAL TABLE nation_s3_json_openx(
n_nationkey string,
n_name string,
n_regionkey string,
n_comment string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://qtest-qubole-com/datasets/presto/functional/nation_s3_json';
;
```

Features

- Entire JSON document must fit in a single line of the text file.
- Read the data stored in JSON format.
- Convert the data to JSON format when INSERT INTO table.
- Arrays and maps are supported.
- Nested data structures are also supported.

Nested JSON Elements

If your data contains nested JSON elements like this:

```
{"country": "Switzerland", "languages": ["German", "French", "Italian"], "religions": {"catholic": [10, 20], "protestant": [15, 25], "orthodox": [5, 15]}}
```

You can declare *languages* as an *array<string>* and *religions* as a *map<string, array<int>* like this (location omitted).

```
CREATE EXTERNAL TABLE json_nested_test (
    country string,
    languages array<string>,
    religions map<string, array<int>>)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
STORED AS TEXTFILE
LOCATION 's3://...'
;
```

You can access a nested element like this

```
select religions['catholic'][0] from json_nested_test;
```

Which produces the result

```
10
```

2.4.6 RCFile Tables

For best performance, we recommend you create RCFiles with binary serialization and using snappy codec to compress data.

Text Serialization of Columns

```
create external table nation_s3_rcfile
(N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
STORED AS RCFILE
LOCATION  's3://qtest-qubole-com/datasets/presto/functional/nation_s3_rcfile';
;
```

Binary Serialization of Columns

```
create external table nation_s3_rcfile
(N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.columnar.LazyBinaryColumnarSerDe'
STORED AS RCFILE
LOCATION  's3://qtest-qubole-com/datasets/presto/functional/nation_s3_rcfile';
;
```

Compression

To compress the data while loading into an RCFile table, use the following *set* statements before inserting data.

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.type=BLOCK;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;
insert into nation_s3_rcfile select * from nation;
```

2.4.7 ORC Tables

Optimized Row Columnar file format offers an efficient way for storing Hive data.

DDL

```
create external table nation_s3_orc
(N_NATIONKEY INT, N_NAME STRING, N_REGIONKEY INT, N_COMMENT STRING)
STORED AS ORC
LOCATION  's3://qtest-qubole-com/datasets/presto/functional/nation_s3_orc'
TBLPROPERTIES ("orc.compress"="SNAPPY")
; 
```

2.4.8 Avro Tables

Qubole supports creating Hive tables against data in [Avro](#) format.

Getting Avro schema from a file

If you have your own avro file, you can extract the schema using avro tools. Download [avro-tools-1.7.4.jar](#) and then run the following command to produce the schema. This schema will go into the serdeproperties in your DDL statement.

```
$ java -jar avro-tools-1.7.4.jar getschema episodes.avro
{
  "type" : "record",
  "name" : "episodes",
  "namespace" : "testing.hive.avro.serde",
  "fields" : [ {
    "name" : "title",
    "type" : "string",
    "doc" : "episode title"
  }, {
    "name" : "air_date",
    "type" : "string",
    "doc" : "initial date"
  }, {
```

```
        "name" : "doctor",
        "type" : "int",
        "doc" : "main actor playing the Doctor in episode"
    } ]
}
```

DDL

The above statement will create a hive table called episodes against avro data. You can then query the table much like any other hive table.

```
CREATE EXTERNAL TABLE episodes
ROW FORMAT
SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
WITH SERDEPROPERTIES ('avro.schema.literal'='
{
    "type" : "record",
    "name" : "episodes",
    "namespace" : "testing.hive.avro.serde",
    "fields" : [ {
        "name" : "title",
        "type" : "string",
        "doc" : "episode title"
    }, {
        "name" : "air_date",
        "type" : "string",
        "doc" : "initial date"
    }, {
        "name" : "doctor",
        "type" : "int",
        "doc" : "main actor playing the Doctor in episode"
    } ]
}
')
STORED AS
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.avro.AvroContainerInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.avro.AvroContainerOutputFormat'
LOCATION 's3://public-qubole/datasets/avro/episodes'
;
```

2.4.9 Hive Bootstrap

Hive Bootstrap is useful when every hive query that is run inside an account needs to perform a series of steps such as:

- Adding jars
- Defining temporary functions
- Setting Hive parameters
- Mapreduce settings for Hive queries

The Hive bootstrap is run before each query is submitted to QDS.

For example, to use test.py in all sessions, define a bootstrap as shown below.

```
add file s3n://prod.qubole.com/ec2-user_hu_6/scripts/test.py;
```

Hive bootstrap can be defined in two ways:

- **Base Bootstrap Location:** It is an S3 location that contains the bootstrap file. See [Adding a Bootstrap File in a Base Bootstrap Location](#) for more information.
- **Bootstrap override editor:** It can be used to manually write/edit entries. See [Adding Settings in the Bootstrap Override Editor](#) for more information.

Adding a Bootstrap File in a Base Bootstrap Location

A file with bootstrap entries is uploaded to an S3 location (for example, mybucket/scripts/hive/base_bootstrap). The location is referred to as base bootstrap location that can be used by multiple accounts.

Base Bootstrap is useful when you want to use the same bootstrap configuration in multiple accounts. If the bootstrap file in S3 is updated, it affects all accounts that use this file.

The base bootstrap location is set at the [Hive Bootstrap in Control Panel](#).

Refer to [Managing Hive Bootstrap](#) for more information on how to set a base bootstrap location.

Adding Settings in the Bootstrap Override Editor

The bootstrap override editor can be set at the [Hive Bootstrap in Control Panel](#). Settings in the bootstrap override editor override the settings in the base bootstrap location. Hence, account-specific settings are added in the bootstrap override editor.

Refer to [Managing Hive Bootstrap](#) for more information on how to set a bootstrap using the bootstrap override editor.

2.4.10 Enabling Hive on Cluster Master

To get Hive to access data in a VPC using its storage handlers, you can run Hive on the cluster master in the VPC. This enables Hive to directly interact with the datasource. You can specify in each Hive query whether you want Hive to run on the cluster master or on its default destination.

Each query that is run on the cluster master must have this configuration:

```
set hive.on.master=true
```

With this setting, the query is redirected to be run on the cluster master. This setting gets removed from the query before submitting to Hive.

Note: However, this setting cannot be added in the Hive bootstrap file. Currently, this feature is supported on Hadoop-1 and Hadoop-2 clusters.

2.4.11 Sessions

Hive allows users to embed their code (python scripts, shell scripts, Java functions) into SQL queries. This is useful as a way to quickly enhance the language to support functionality that is not natively present in HiveQL. See [here](#) and [here](#) for examples of SQL queries using this functionality.

Qubole mimics this functionality by associating every command in QDS with a session. Sessions allow you to create temporary data sets (that last for the duration of the session) as well as configure specific session parameters which can be used to tune query behavior. Sessions also allow you to add your code as scripts to Qubole so that they could be used to run transformations within HiveQL. Through sessions you can isolate the effects of these parameters,

temporary datasets and user defined transformations across your various workloads. Sessions have lifetimes that can be hand-edited (default value 2 hours).

To ensure that the scripts are accessible to our Hive clusters upload them to Amazon's S3 storage service. The script can be in any S3 location that is readable using the keys associated with account. However we recommend that you place all your scripts in a scripts folder under your default location. For example, if your default location is prod.qubole.com/ec2-user_hu_6 we suggest putting any scripts under s3://prod.qubole.com/ec2-user_hu_6/scripts/. Once a script has been uploaded to a S3 location you can add it to your session by running a "add file ..." command. This command will be automatically added to the session if it is successful and this script is then usable in further queries that are part of this session. For e.g. consider a simple test.py script which has the following code:

test.py

```
#!/usr/bin/python

import sys
line = sys.stdin.readline();
while line:
    print line
    line = sys.stdin.readline();
sys.exit(0)
```

If the default location is prod.qubole.com/ec2-user_hu_6 then upload test.py to prod.qubole.com/ec2-user_hu_6/scripts/test.py. Once this has been uploaded you would issue:

```
add file s3n://prod.qubole.com/ec2-user_hu_6/scripts/test.py;
//and then use this in a query as
select count (*)
from (select transform(a) using 'python test.py' from tb) V
```

2.5 Hadoop in Qubole

Qubole runs applications written in Map-Reduce, Cascading, Pig, Hive, and Scalding using Apache Hadoop. Qubole offers two flavors of Hadoop based on Apache Hadoop 1.x and 2.x releases that are compatible with open source APIs and documentation regarding the same. Qubole has provided some additional controls to the Hadoop features that are relevant to advanced users.

The following sections cover the features associated with Hadoop 1.x and Hadoop 2.x that are relevant to advanced users:

2.5.1 Hadoop 1 in Qubole

This section covers topics associated with Hadoop 1.x that are relevant to advanced users. Qubole has provided some additional controls to the Hadoop features.

Configuring Fair Scheduler on Hadoop

To configure Fair Scheduler in Qubole on Hadoop 0.20.1, perform the following steps:

1. Create a XML file for the Fair Scheduler configuration. A sample Fair Scheduler XML is as provided below.

```
<allocations>
<pool name="default">
  <minMaps>72</minMaps>
  <canBePreempted>true</canBePreempted>
```

```

<fifo>false</fifo>
</pool>
<pool name="sqoop-conf">
  <maxMaps>32</maxMaps>
  <maxReduces>8</maxReduces>
  <fifo>true</fifo>
</pool>
<pool name="sqoop-conf2">
  <maxMaps>2</maxMaps>
  <fifo>true</fifo>
</pool>
<pool name="batch">
  <minMaps>216</minMaps>
  <fifo>true</fifo>
</pool>
<pool name="fast-paced">
  <minMaps>50</minMaps>
  <fifo>false</fifo>
  <minSharePreemptionTimeout>300</minSharePreemptionTimeout>
</pool>
<fairSharePreemptionTimeout>60</fairSharePreemptionTimeout>
</allocations>

```

Within a <pool> tag, <canBePreempted>true</canBePreempted> is an important XML tag. By default, the value defaults to true and all pools are preemptible. You can set an individual pool to *false*, to prevent tasks from that pool from being preempted.

Use <canBePreempted>true</canBePreempted> with caution because this can allow tasks from this (non-preemptible) pool to hog the cluster. It makes sense to use the tag for small jobs that do important activity.

The last element, <fairSharePreemptionTimeout>60</fairSharePreemptionTimeout> means by default tasks are preempted after 60s (if they do not get their fair share of the cluster). This is something you frequently want to change to see how aggressive preemption is.

2. The Fair Scheduler configuration is at the cluster level that is available in the **Edit Cluster** tab of the **Control Panel** page.
3. The Fair Scheduler configuration can include certain custom XML tags that are specific to Qubole.

Example: Qubole specifies use of FIFO scheduling within a pool differently than open source Hadoop.

Use <fifo>true</fifo> instead of <schedulingMode>FIFO</schedulingMode> (applicable to later versions of Hadoop Fair Scheduler).

This property is specified within a Fair Scheduler pool. It denotes whether the resources within a pool are allocated to jobs in a FIFO way. If it is set to *false*, resources within a pool are allocated fairly.

4. If you restart the cluster or **Push** the configuration (**Push** is available as one of the actions in the **Action** column of the **Clusters** tab in the **Control Panel** page of the QDS UI), the new Fair Scheduler settings become effective. See *Cluster Operations* for more information.
5. When a job is being submitted, you can specify which Fair Scheduler pool to use. The specified Fair Scheduler pool overrides the default Fair Scheduler pool specified (if any). When you run database import/export commands, you can select a Fair Scheduler pool from the QDS UI. If you are running *sqoop* from a command line interface (through a Shell command), you can use the *-Dmapred.fairscheduler.pool* switch.
6. **Small jobs limit**

In Fair Scheduler, Qubole allows *small* jobs to go through even if the system is burdened with many big jobs.

A mapreduce job is a *small* job if it has a maximum of 10 tasks or less than (*mapred.fairscheduler.smalljob.taskperc * totalTaskLimit*) tasks.

Where:

mapred.fairscheduler.smalljob.taskperc is a Hadoop configuration (float) and must be configured in the value range of (0 - 1).

totalTaskLimit is *mapred.fairscheduler.total.task.limit* - It denotes the maximum number of jobs permitted by the JobTracker (default value is 800000).

7. Auto task limit

Qubole uses a smarter way to calculate the total task limit based on the machine specifications on which the JobTracker (JT) runs. It is enabled by setting this parameter, *mapred.fairscheduler.auto.task.limit.enable* (default parameter value is true).

Total task limit in this case is: (20000 * heap_memory of the JT)

You can change the value, 20000 by using the parameter, *mapred.fairscheduler.auto.task.limit.pergb*.

Protecting Clusters Against Bad Jobs

A single hadoop cluster is usually shared across many users. In such a multi-user environment, it is a common occurrence that a certain user may issue a bad job which may degrade the performance of the entire cluster. Some common occurrences of bad jobs affecting the whole clusters are

- Single mapper issuing too much output thereby taking away the disk with it
- A map/reduce job may have a lot of mappers outputting too much of map data thereby affecting the disk usage of the entire cluster
- Reducer tasks copying a lot of map output data during the shuffle phase thereby affecting the disk space

Qubole's hadoop distribution provides protection of the clusters against such jobs. Its clusters periodically monitors the jobs (via the JobUpdater thread) and kills any job that may be affecting the entire cluster.

Cluster Level Configuration

Following are the cluster level parameters and their default values. For changing them, add them to the hadoop overrides section and restart the cluster.

Control the frequency of JobUpdater thread

```
mapred.cluster.kill.jobs.map.output.interval.seconds (default: 600 seconds)
```

Kill the job producing maximum map output when DFS is running low on disk space

```
# This function will be invoked when average space per node is less than the
# configured value
mapred.hustler.downscale.freespacepernode.gb.avg (default: 5)

# And when the map output is more than the following fraction of the used DFS.
mapred.map.output.cluster.capacity.ratio.max (default=0.5)
```

Job Level Configuration

Following are the job level configuration properties and their default values. These values can either be changed by modifying them in the hadoop overrides (cluster restart is not required) or changing them on a per job basis.

Kill job which produces too much map output

```
# Kill the job when the total map output is beyond the configurable value
mapred.map.output.job.max.gb (default: 20% of the total configured dfs capacity)

# Kill the job when any of its tasks produces more map output than the configurable value
mapred.map.output.task.max.gb (default: 80% of the disk space per node)
```

Kill job which produces a lot of logs

```
# Kill the job when any of its tasks produces logs with size more than the
# configurable value
mapred.task.log.size.max (default: 40% of the disk space per node)
```

Kill job where reducers read a lot of map data

```
# Kill the job when any of the reduce tasks reads more shuffle data than the
# configurable value
mapred.reduce.shuffle.bytes.per.task.max.gb (default: 80% of the disk space per node)
```

Developing MapReduce Applications on Qubole

Qubole has uploaded its Hadoop jars into a maven repository, which can be used to develop MapReduce applications to ensure compatibility with Qubole. Sample build files for Hadoop 1 and Hadoop 2 are provided in [Qubole Jar Test](#) along with example projects.

Note: When you extract files from an Hadoop archive, by default, files get extracted to a folder with a name that matches exactly with the Hadoop archive's name. For example, files from the **user/zoo/test.tar** archive are extracted into the **user/zoo/test.tar** folder.

Locating Logs on S3

On Amazon S3:

The Hadoop-1 logs are located at: DEFLOC/logs/hadoop/<timestamp>/.

Where:

- **DEFLOC** refers to the default location of an account.
- <**timestamp**> denotes the starting time of the cluster.

Storage Volume Management

Hadoop uses instance storage for important features such as caching, fault tolerance and storage. Hadoop Distributed File System (HDFS) relies on the local storage on instances within cluster to store replicas of data blocks. MapReduce (MR) framework uses local file system for storing intermediate map outputs, spill files, distributed cache etc. which can result in high volumes of disk usage while working with reasonably sized datasets.

Hadoop uses volumes attached to an instance in round-robin fashion to distribute files across different disks. A task attempt writing to a volume with no available storage space can fail. The task attempt will normally be retried but such failures can lead to job failure if maximum retry attempts are exceeded.

Volumes and Task Configuration Parameters

Some of the important Hadoop configuration properties which can be used to avoid out of disk space errors are:

Parameter	De-fault Value	Description
dfs.datanode.du.reserved	1073741824	Reserved space in bytes per volume. On QDS, both MR and HDFS will not select a volume for write if it has available space less than the configured value. Value in bytes.
mapred.local.dir	2147483648	If no volume on the instance has available space more than the configured value, new tasks are not scheduled on the instance. Value in bytes.
mapred.local.dir.0.minspace	0	If no volume on the instance has available space more than the configured value, new tasks are not scheduled on the instance. Also, to save the rest of the running tasks, kill one of them, to clean up some space. Start with the reduce tasks, then go with the ones that have finished the least. Value in bytes.

EBS Volumes for Additional Storage

Some instances on cloud, like the compute intensive AWS instances, have good CPU and memory configuration but come with low instance storage which limits their usage. To increase storage on such instance types, it is possible to attach EBS volumes using cluster configuration in the **Control Panel**. Qubole Data Service (QDS) allows both EBS Magnetic and EBS General Purpose (SSD) volumes to be attached.

There are two modes in which EBS volumes can be used:

- **EBS Volumes as Reserved Disks**

In this mode, the attached EBS volumes act as reserved disks for HDFS and MR intermediate data. An additional configuration property `local.reserved.dir` is automatically set to contain list of directories residing on mounted EBS volumes. These directories are accessed only in case the request to read, write or existence check could not be served from instance storage.

The above arrangement ensures that if there was sufficient space on fast SSD volumes, then EBS volumes are not used. This prevents the use of EBS volumes which might have higher latency or have I/O operation cost associated with them. Additional information related to this mode can be found in the [blog post](#). This is the default mode in which the attached EBS volumes are used.

- **EBS Volumes as Local Disks**

In this mode, the attached EBS volumes act as regular instance storage. The mode can be enabled by setting Hadoop configuration property `local.reserved.regular.disk` to `true`. This mode is useful for cases when SSD volumes are attached to the instance which have desired latency for the application and do not have additional I/O cost associated with them.

Manually Scaling a Hadoop Cluster

Manual scaling provides explicit cluster control for manually adding nodes to a Hadoop-1 cluster. Manual scaling is useful when:

- You want to add more nodes than the Qubole's auto-scaling limit to process a lot of reduce tasks that are running.
- You want to scale up beyond the maximum limit specified in the configuration.

Qubole supports an add node API to manually scale a Hadoop cluster. Currently, this API only supports Hadoop 1 clusters and HBase. See [Add a Node to a Cluster](#) for more information on the REST API.

Speculative Execution

In MapReduce a job is broken into several tasks which will execute in parallel. This model of execution is sensitive to slow tasks (even if they are very few in number) as they will slowdown the overall execution of a job. Therefore, Hadoop detects such slow tasks and runs (duplicate) backup tasks for such tasks. This is called *speculative execution*. Speculating more tasks can help jobs finish faster - but can also waste CPU cycles. Conversely - speculating fewer tasks can save CPU cycles - but cause jobs to finish slower. The options documented here allow the users to control the aggressiveness of the speculation algorithms and choose the right balance between efficiency and latency.

All the options below apply on and are settable on a *per-job* level. Indicative values are usually defaults.

Speculation Quota

There is a hard limit of 10% of slots used for speculation across *all* hadoop jobs. This is not configurable right now. However there is a per-job option to cap the ratio of speculated tasks to total tasks:

```
mapreduce.job.speculative.speculativecap=0.1
```

The value of above option can be any floating number between 0.01 and 1. It cannot be set below 0.01. By default the value is 0.1 i.e., 10% of tasks can be speculated at any time. Note that the ratio is applied to map and reduce tasks separately.

Controls on Speculation of Individual Tasks

Following are the controls on speculative execution of individual tasks and are applied in the order below:

1. A task **is not** speculated for the first 60 seconds by default. It is overridable on per-job level via the following options:

```
mapred.speculative.map.lag=60000
```

```
mapred.speculative.reduce.lag=60000
```

2. A task that is projected to finish soon **is not** speculated. By default this feature is disabled. To enable, set the following options:

```
mapred.speculative.map.duration=300
```

```
mapred.speculative.reduce.duration=300
```

In this example setting, if a task is projected to take 300 seconds or less then it is not speculated.

3. A task whose progress rate is low compared to other tasks *is* speculated.

The progress rate of each task is compared to the mean of all the other tasks in the job. If the difference is more than the standard deviation, then it is speculated. Speculation can be made more or less aggressive by comparing the difference to a multiple of the standard deviation that is controlled by the setting below:

```
mapreduce.job.speculative.slowtaskthreshold=1.0
```

This value is 1.0 by default. In some cases - the standard deviation is often very large (and this results in no tasks being speculated). As a result the standard deviation is capped to a certain multiple of the mean that is controlled by the setting below (with the default value of 0.8 in Qubole):

```
mapreduce.job.speculative.stddevmeanratio.max=0.8
```

4. If a task belongs to the last few mappers or reducers - then it **is** speculated (regardless of its progress rate). This has been found to be extremely useful in taking care of lone straggler reducers and mappers.

- This feature is enabled by default in Qubole and is controlled by:

```
mapred.reduce.tasks.speculation.unfinished.threshold=0.001
```

```
mapred.map.tasks.speculation.unfinished.threshold=0.001
```

- With the above values the last 0.1% of tasks or 1 task (minimum) are always speculated. Most often you will see the last task being speculated.
- To disable this feature - set the above options to value of 0.

Strategies for Dealing with Out of Memory Errors

Jobs can sometimes fail due to their tasks running out of heap memory. Based on where these out of memory errors occur, we suggest few configuration tunings to deal with them.

Out of Memory Errors on Mapper Side

If out of memory errors are occurring on the mapper side, then reduce the value of following configuration:

```
#The maximum number of streams to merge at once when sorting files.  
io.sort.factor (default: 10)
```

Out of Memory Errors on Reducer Side

If out of memory errors are occurring on the reducer side, then check if errors are thrown in:

- Copy/Shuffle phase

In this case, reduce the value of following two configurations:

```
#The number of threads used to copy map outputs to the reducer.  
mapred.reduce.parallel.copies (default: 5)
```

```
#The proportion of total heap size to be allocated to the map outputs buffer  
#during the copy phase of the shuffle.  
mapred.job.shuffle.input.buffer.percent (default: 70% of the total heap size)
```

- Merge phase

In this case, reduce the value of following configuration:

```
#The maximum number of streams to merge at once when sorting files.  
io.sort.factor (default: 10)
```

2.5.2 Hadoop 2 in Qubole

This section covers topics associated with Hadoop 2.x that are relevant to advanced users. Qubole has provided some additional controls to the Hadoop features.

MapReduce Configuration in Hadoop 2

Qubole's Hadoop 2 offering is based on Apache Hadoop 2.6.0. Qubole has some optimizations in S3 access and has enhanced it with its autoscaling code. Qubole jars have been uploaded in a maven repository and can be accessed seamlessly for developing mapreduce/yarn applications as highlighted by this [POM](#) file.

One of the significant shifts from Hadoop 1 to Hadoop 2 is the JobTracker's functionality split into **Resource Manager** and **Application Master**. The other significant shift is the replacement of **Map** and **Reduce** slots by **containers**.

In Hadoop 1, a task tracker (slave node) has a maximum limit on the number of map and reduce slots. This means that if a node is running the limit on the map slots, more map slots cannot be scheduled even when more resources are available.

In Hadoop 2, slots have been replaced by containers, which is an abstracted part of the slave resources. A container can be of any size within the limit of the Node Manager (slave node). The map and reduce tasks are Java Virtual Machines (JVMs) launched within these containers.

This change means that specifying the container sizes become important. For example, a memory-heavy map task, would require a larger container than a lighter map task. Moreover, the container sizes are different for different instance types (for example, an instance with larger memory has larger container size). While Qubole specifies good default parameters for the container sizes per instances, there are certain cases when you would like to change the defaults.

The default Hadoop 2 settings for a cluster is shown in the **Edit Cluster** page of a Hadoop 2 Cluster. (Navigate to Control Panel > Clusters and click the edit button that is against a Hadoop 2 Cluster. See [Managing Clusters](#) for more information.) A sample is as shown in the following figure.

Hadoop Cluster Settings

Override Hadoop Configuration Variables	<pre>mapreduce.map.memory.mb=2240 mapreduce.reduce.java.opts=-Xmx2016m mapreduce.reduce.memory.mb=2240 yarn.app.mapreduce.am.command-opts=-Xmx2016m yarn.scheduler.minimum-allocation-vcores=1 yarn.scheduler.maximum-allocation-vcores=8 yarn.nodemanager.resource.memory-mb=13440 yarn.scheduler.minimum-allocation-mb=1024 yarn.app.mapreduce.am.resource.cpu-vcores=1 mapreduce.reduce.cpu.vcores=1 yarn.nodemanager.resource.cpu-vcores=8 yarn.app.mapreduce.am.resource.mb=2240 mapreduce.map.java.opts=-Xmx2016m yarn.scheduler.maximum-allocation-mb=13440 mapreduce.map.cpu.vcores=1 mapreduce.task.io.sort.mb=896</pre>
---	---

Recommended Configuration

In MapReduce, changing a task's memory requirement requires changing the following parameters:

- The size of the container in which the map/reduce task is launched.
- Specifying the maximum memory (-Xmx) to the JVM of the map/reduce task.

The two parameters (mentioned above) are changed for **MapReduce Tasks/Application Master** as shown below:

- **Map Tasks:**

```
mapreduce.map.memory.mb=2240 # Container size  
mapreduce.map.java.opts=-Xmx2016m # JVM arguments for a Map task
```

- **Reduce Tasks:**

```
mapreduce.reduce.memory.mb=2240 # Container size  
mapreduce.reduce.java.opts=-Xmx2016m # JVM arguments for a Reduce task
```

- **MapReduce Application Master:**

```
yarn.app.mapreduce.am.resource.mb=2240 # Container size  
yarn.app.mapreduce.am.command-opts=-Xmx2016m # JVM arguments for an Application Master
```

Locating Logs on S3

On Amazon S3:

- The YARN logs (Application Master and container logs) are stored at: s3://<DEFLOC>/logs/hadoop/CLUSTER_ID/CLUSTER_INST_ID/app-logs.
- The daemon logs for each host are stored at: s3://<DEFLOC>/logs/hadoop/CLUSTER_ID/CLUSTER_INST_ID/HOST
- The MapReduce Job History files are stored at: s3://<DEFLOC>/logs/hadoop/CLUSTER_ID/CLUSTER_INST_ID/mr-history

Where:

- **DEFLOC** refers to the default location of an account.
- **CLUSTER_INST_ID** is the cluster instance ID. It is the latest folder in the location, s3://DEFLOC/logs/hadoop/CLUSTER_ID/ for a running cluster or the last-terminated cluster.

Significant Parameters in Hadoop 2

The following parameters can be useful in an Hadoop-2 configuration:

- **DirectFileOutputCommitter** used for configuring a direct file output committer for a MapReduce task. See [Configuring Direct File Output Committer](#) for more information.
- **mapreduce.jobhistory.completed.codec** used for compressing job history to store it in an Amazon S3 location. See [Configuring Job History Compression](#) for more information.
- **yarn.scheduler.qubole.am-on-stable.timeout.ms** used for configuring a timeout for a Resource Manager (RM) to schedule Application Masters (AMs) on volatile spot nodes. See [Configuring Timeout to Schedule AMs on Volatile Spot Nodes](#) for more information.

This section describes the purpose of setting the parameters and how to configure them.

Configuring Direct File Output Committer

In general, the final output of a MapReduce job is written to a location in S3 or HDFS. However, each task that is supposed to write the output data into a final location first writes it into a temporary location. It is in a task's *commit phase*, when the output data is moved from the temporary location to a final location.

DirectFileOutputCommitter (DFOC) is a setting, which when enabled lets a task write the output data directly to the final location. In this case, a commit phase is not required. It is a Qubole-specific parameter that is also supported by other big data vendors.

The pros and cons of setting DFOC are as mentioned below.

Pros:

- Setting DFOC avoids Eventual Consistency issues (although Eventual Consistency is not applicable to us-east1 region now).
- Improves performance when data is written into an Amazon S3 location. It does not show much impact on the performance when data is written into a HDFS location because in HDFS, the movement of files from one directory to another directory is very fast. Since Amazon S3 does not support native move operation, the commit phase is longer when large amounts of data is written to an S3 location. The commit phase can be optimized by setting DFOC.

Cons:

- DFOC does not perform well under failures. Stale data may be present in the final location in case of failures and workflows are generally designed to delete the final location in case of a failure. Hence, Qubole does not enable DFOC by default. If DFOC is disabled, then the abort phase of the task deletes the data in the temporary directory and a retry takes care of the data deletion. So, there is no stale data in the final location.

Enabling DFOC DFOC can be set in the two MapReduce APIs, *mapred* and *mapreduce* as mentioned below:

- DFOC in Mapred API:
`mapred.output.committer.class=org.apache.hadoop.mapred.DirectFileOutputCommitter.class`
- DFOC in Mapreduce API: `mapreduce.use.directfileoutputcommitter=true`

Configuring Job History Compression

`mapreduce.jobhistory.completed.codec` is a parameter that is set to compress the job history files while storing them in an Amazon S3 location using this codec. Its default value is:

`com.hadoop.compression.lzo.LzopCodec`.

Configuring Timeout to Schedule AMs on Volatile Spot Nodes

Qubole has introduced **yarn.scheduler.qubole.am-on-stable.timeout.ms** to set a timeout for scheduling AMs on [volatile spot nodes](#). The timeout is set in milliseconds. By default, Qubole does not schedule AMs on volatile spot nodes. This is because such nodes can go away at any time and losing the AM of a YARN application can be disastrous. This default is specified by setting the above parameter to -1.

However, there may be cases where you can run AMs on such volatile spot nodes, for example, when you are really cost conscious and a cluster primarily contains volatile spot nodes. In that case, you can use this parameter to set a timeout. RM tries to schedule AMs on stable nodes first, however, once the timeout is hit and the RM has not been able to schedule the AM, it considers volatile spot nodes as well.

When you set the parameter's value to 0, RM immediately considers all nodes when trying to schedule the AM.

This parameter can have the values as described in the following table.

Parameter Values	Description
-1	It is the default value that implies that AMs are not scheduled on volatile spot nodes.
0	RM schedules AMs on volatile spot nodes whenever possible (after waiting for 0 millisecond).
Any other value	RM waits for <value> milliseconds before scheduling an AM on a volatile spot node.

Configuring Job Runtime Use `yarn.resourcemanager.app.timeout.minutes` to configure the duration in minutes for an YARN application to run. This parameter can prevent runaway applications (if any) from keeping the cluster alive unnecessarily.

It is a cluster-level configuration and must be set in the **Override Hadoop Configuration Variables** available in the **<Cluster Configuration> > Hadoop Cluster Settings** user interface. See *Modifying Hadoop Cluster Settings* for more information.

ResourceManager kills an YARN application if it is running for longer than the configured timeout.

ResourceManager never kills an YARN application based on the timeout if it is set to **-1**.

2.6 HBase in Qubole

2.6.1 Introduction to HBase

Qubole's HBase is based on Apache HBase1.0.0 on Hadoop2 2.6.0.

HBase clusters are used when there is a need for random and realtime read/write access to Big Data. HBase is an open-source, distributed, versioned, and non-relational database modeled after Google's Bigtable. HBase provides Bigtable-like functionality on top of Hadoop and HDFS.

The features of Qubole HBase are:

- Cluster monitoring and alerts
- Cluster management
- Snapshots

2.6.2 Configuring HBase Clusters



Navigate to Control Panel and in the **Clusters** tab, click the add icon that is on the top-right corner of the **Clusters** page to add a cluster.

The **Add New Cluster** page is displayed. Enter the following details to create a new cluster:

- Configuring Cluster Settings. See *Configuring Cluster Settings*.
- Configuring EC2 Settings. See *Configuring EC2 Settings*.
- Configuring HBase Cluster Settings. See *Configuring HBase Cluster Settings*.
- Configuring Security Settings. See *Configuring Security Settings*.

Configuring Cluster Settings

- **Cluster Labels:** A cluster can have one or more labels separated by a comma. You can set/label a cluster as default by selecting the default option. This is a mandatory field.
- **Cluster Type:** It shows the type of cluster. By default, **Hadoop** is the default cluster type on QDS. **Hadoop2**, **Presto**, **Spark**, and **HBase** are the other cluster types. Select **HBase** as the cluster type. The HBase cluster settings page is displayed as shown in the following figure.

Cluster > Add New Cluster

Cluster Information							
Cluster Settings							
Cluster Labels	<input type="text"/>						
Cluster Type	HBase						
Region Server Count	<input type="text"/>						
Master Node Type	m1.large - 2 cores, 7.5GiB m						
Slave Node Type	m1.xlarge - 4 cores, 15GiB m						
Node Bootstrap File	s3://data.com/logs/production/scripts/hadoop/ node_bootstrap.sh						
Other Settings	<input checked="" type="checkbox"/> Enable Ganglia Monitoring						
EC2 Settings (Sample IAM Credentials)							
Compute Access Key	<input type="text"/>						
Compute Secret Key	<input type="text"/> Hide Secret Key						
Region	us-east-1						
Aws Availability Zone	No Preference						
VPC	None (optional)						
Subnet	(optional)						
Custom EC2 Tags	<table border="1"> <thead> <tr> <th>Custom Tag</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td><input type="text"/></td> <td><input type="text"/> x</td> </tr> <tr> <td colspan="2">+ Add EC2 Tags</td> </tr> </tbody> </table>	Custom Tag	Value	<input type="text"/>	<input type="text"/> x	+ Add EC2 Tags	
Custom Tag	Value						
<input type="text"/>	<input type="text"/> x						
+ Add EC2 Tags							
Hbase Cluster Settings							
Override Hbase Configuration Variables	<input type="text"/>						
Recommended Configuration	<input type="text"/>						
Security Settings							
Qubole Public Key	ssh-rsa AAAQABAAQABAAAAAAAADQABAAABAAAAAAAAGlraAAAAAvLGel						
Customer Public ssh Key	<input type="text"/> (optional)						
Persistent security groups	<input type="text"/> (optional)						
<input type="button" value="Save"/> <input type="button" value="Cancel"/>							

- **Region Server Count** - It is the count of number of slave nodes that a HBase cluster can contain. Slave nodes are referred to as **Region Servers**.
- **Master Node Type**: Set the master node type by selecting the preferred node type from the drop-down list.
- **Slave Node Type**: Set the slave node type by selecting the preferred node type from the drop-down list. However, if you select the **c3**, **m3**, and **r3** slave node type, you get three additional Elastic Block Storage (EBS) configurations, that is **EBS Volume Count**, **EBS Volume Type**, and **EBS Volume Size** (supported range is 100 - 500 GB).
- **Node Bootstrap File**: Set the node bootstrap file.
- **Other Settings**: By default, **Enable Ganglia Monitoring** is enabled.

Configuring EC2 Settings

Amazon Elastic Cloud Compute (EC2) web service offers resizable compute capacity in the cloud. EC2 settings' fields are visible only when the **Compute Type** is **CUSTOMER_MANAGED**.

EC2 Settings (refer to the above figure) contains the following options:

- **Compute Access Key** - Enter an access key (along with the secret key) to set AWS Credentials for launching the cluster.
- **Secret Key** - Enter a secret key (along with the access key) to set AWS Credentials for launching the cluster. Click **Edit Secret Key** to add a different key in the textfield as shown in the following figure.

EC2 Settings

Same As Default Compute	<input type="checkbox"/>						
Compute Access Key	ABCDEFGHIJKLMNPQRST						
Compute Secret Key	***** Edit Secret Key						
Region	us-east-1  						
Aws Availability Zone	No Preference   						
VPC	None  (optional) 						
Subnet	 (optional)						
Custom EC2 Tags	<table border="1"> <thead> <tr> <th>Custom Tag</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Qubole</td> <td>qbol_acc1</td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table> + Add EC2 Tags	Custom Tag	Value	Qubole	qbol_acc1		
Custom Tag	Value						
Qubole	qbol_acc1						
							

Click **Hide Secret Key** after adding a key to hide it.

- **AWS Region** - Denotes the region from which the nodes are launched. The default region is *us-east-1*. Select a region from the drop-down list to change it.
- **AWS Availability Zone** - By default, the availability zone is set to *No Preference* and the best availability zone based on system health and available capacity is selected. Note that all cluster nodes are in the same availability

zone. You can select an availability zone from the drop-down list, in which instances are reserved to benefit from the AWS reserved instances. Click the refresh icon  to refresh the availability zones' list.

- **VPC** - Set this configuration to launch the cluster within an Amazon Virtual Private Cloud (VPC). It is an optional configuration. If VPC is not specified, cluster nodes are placed in a Qubole created Security Group in EC2-Classic or in the *default VPC* for EC2-VPC accounts. By default, **None** is selected. Click the refresh icon  to refresh the VPC list.
- **Subnet** - When you select a VPC, a corresponding subnet gets selected. By default, the textfield is blank as no VPC is selected.
- **Custom EC2 Tags** - You can set a custom EC2 tag if you want the instances of a cluster to get that tag on AWS. This tag is useful in billing across teams as you get the AWS cost per tag, which helps to calculate the AWS costs of different teams in a company.

Add a tag in **Custom Tag** and enter a value for the tag in the corresponding text field as shown in the above figure. Tags and values must have alphanumeric characters and can contain only these special characters: + (plus-sign), . (full stop/period/dot), - (hyphen), @ (at-the-rate of symbol), and _ (an underscore). The tags, *Qubole*, *name*, and *alias* are reserved for use by Qubole. Tags beginning with *aws-* are reserved for use by Amazon. Click **Add EC2 tags** if you want to add more than one custom tag. A maximum of five custom tags can be added per cluster.

EC2 Settings provides more information. See [Manage Roles](#) for information on how permissions to access features are granted and restricted.

Configuring HBase Cluster Settings

In **HBase Cluster Settings**, you can:

- Override the Hbase configuration by entering the variables in the **Override Hbase Configuration Variables** text field.
- Use **Recommended Configuration** for the selected slave instance type unless it is overridden.

Configuring Security Settings

Qubole Public Key cannot be changed. Setting **Customer Public SSH Key** is optional.

Persistent security groups - This option overrides the account-level security group settings. By default, this option is not set but it inherits the account-level persistent security group, if any. Use this option if you want to give additional access permissions to cluster nodes.

After configuring the various options of a cluster, click **Save**. Click **Cancel** to not save the settings.

Note: You can create, start, edit, terminate, and delete a HBase cluster. See [Cluster Operations](#) for more information.

2.6.3 Editing HBase Cluster Configuration

Navigate to Control Panel and in the **Clusters** tab. The clusters are as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)					
Search : Enter Search Text <input type="button" value="X"/>							
Id	Labels		Nodes	Up Time	Resources	Action	
● 4032 ✓	hadoop1	default	0		Cluster Usage Report	  	
● 4033	hadoop2		0		Cluster Usage Report	  	
● 4034	presto		0		Cluster Usage Report	  	
● 4035	spark		0		Cluster Usage Report	  	
● 4131	hbase		2(0)	a few seconds	Cluster Usage Report View Master DNS Hannibal HBase HBase shell HBase Master Ganglia Metrics	  	

Click the edit icon  available in the **Action** column to edit a HBase cluster.

Note: You can create, start, edit, terminate, and delete a HBase cluster. See [Cluster Operations](#) for more information.

HBase cluster settings are displayed as shown in the following figure.

Cluster > Edit Cluster (6920)

Cluster Information

Cluster Settings

Cluster Labels	<input type="text" value="hbase_test_workable"/>
Cluster Type	<input type="text" value="HBase"/>
Region Server Count	<input type="text" value="2"/>
Master Node Type	<input type="text" value="m1.large - 2 cores, 7.5GiB mem"/>
Slave Node Type	<input type="text" value="m1.medium - 1 cores, 3.75GiB mem"/>
Node Bootstrap File	<input type="text" value="s3://data.com/logs/production/scripts/hadoop/bootstrap.sh"/>
Other Settings	<input checked="" type="checkbox"/> Enable Ganglia Monitoring

EC2 Settings [\(Sample IAM Credentials \)](#)

Compute Access Key	<input type="text" value="ABCDEFGHIJKLMNPQRSTUVWXYZ"/>								
Compute Secret Key	<input type="text" value="*****"/> Edit Secret Key								
Region	<input type="text" value="us-east-1"/>								
Aws Availability Zone	<input type="text" value="No Preference"/>								
VPC	<input type="text" value="None"/> (optional)								
Subnet	<input type="text"/> (optional)								
Custom EC2 Tags	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">Custom Tag</th> <th style="text-align: left;">Value</th> </tr> </thead> <tbody> <tr> <td><input type="text" value="Qubole"/></td> <td><input type="text" value="qbol_acc1"/> </td> </tr> <tr> <td><input type="text"/></td> <td><input type="text"/> </td> </tr> <tr> <td colspan="2" style="text-align: center;">+ Add EC2 Tags</td> </tr> </tbody> </table>	Custom Tag	Value	<input type="text" value="Qubole"/>	<input type="text" value="qbol_acc1"/>	<input type="text"/>	<input type="text"/>	+ Add EC2 Tags	
Custom Tag	Value								
<input type="text" value="Qubole"/>	<input type="text" value="qbol_acc1"/>								
<input type="text"/>	<input type="text"/>								
+ Add EC2 Tags									

Hbase Cluster Settings

Override Hbase Configuration Variables	<input type="text" value="mapred.job.hustler.enabled=true"/>
Recommended Configuration	<input type="text"/>

Security Settings

Qubole Public Key	<input type="text" value="ssh-rsa AAAA...AgLrAAAAAvLGef"/>
Customer Public ssh Key	<input type="text"/> (optional)
Persistent security groups	<input type="text"/> (optional)

[Save](#)

[Cancel](#)

All the fields can be edited. See [Configuring HBase Clusters](#) for more information on the configuration settings.

After modifying the various options of a cluster, click **Save**. The edited settings apply to the cluster only after it is

restarted. Click **Cancel** to restore the previous settings.

2.6.4 Monitoring an HBase Cluster

Qubole provides Hannibal HBase, which is an interface to monitor an HBase cluster, HBase Master to see the master and region server details, and a HBase shell to run queries that are explained in the following topics:

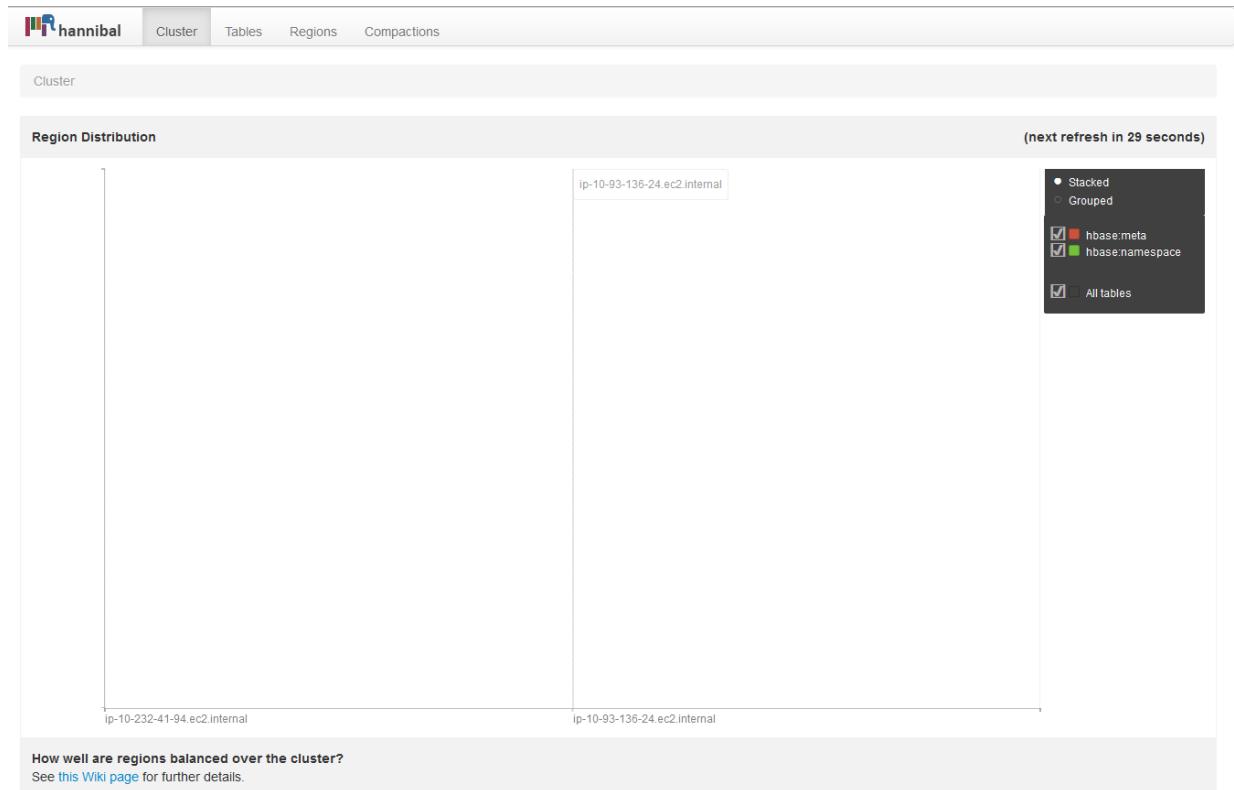
- *Viewing Hannibal HBase*
- *Viewing HBase Master*
- *Using HBase Shell*

Viewing Hannibal HBase

Hannibal HBase is listed in **Resources** of a running HBase cluster as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)						
Search : Enter Search Text						+	↻	
Id	Labels	Nodes	Up Time	Resources	Action			
● 4032 ✓	hadoop1 default	0		Cluster Usage Report				
● 4033	hadoop2	0		Cluster Usage Report				
● 4034	presto	0		Cluster Usage Report				
● 4035	spark	0		Cluster Usage Report				
● 4131	hbase	2(0)	a few seconds	Cluster Usage Report View Master DNS Hannibal HBase HBase shell HBase Master Ganglia Metrics				

Click **Hannibal HBase** and a new interface is displayed as shown in the following figure.



Viewing HBase Master

HBase Master is listed in **Resources** of a running HBase cluster as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)				
		Search : Enter Search Text				
Id		Labels	Nodes	Up Time	Resources	Action
●	4032	hadoop1 default	0		Cluster Usage Report	
●	4033	hadoop2	0		Cluster Usage Report	
●	4034	presto	0		Cluster Usage Report	
●	4035	spark	0		Cluster Usage Report	
●	4131	hbase	2(0)	a few seconds	Cluster Usage Report View Master DNS Hannibal HBase HBase shell HBase Master Ganglia Metrics	

Click **HBase Master** from the drop-down list. The **HBase Master** page is displayed as shown in the following figure.

The screenshot shows the Qubole Data Service HBase Master interface. At the top, there's a navigation bar with links: Home, Table Details, Local Logs, Log Level, Debug Dump, Metrics Dump, and HBase Configuration. Below the navigation bar, it says "Master ip-10-232-21-156.ec2.internal".

Region Servers

ServerName	Start time	Requests Per Second	Num. Regions
ip-10-33-131-49.ec2.internal,16020,1439533450359	Fri Aug 14 06:24:10 UTC 2015	0	0
ip-10-93-136-24.ec2.internal,16020,1439533447637	Fri Aug 14 06:24:07 UTC 2015	0	2
Total:2		0	2

Backup Masters

ServerName	Port	Start Time
Total:0		

Tables

User Tables	System Tables	Snapshots

Tasks

Show All Monitored Tasks Show non-RPC Tasks Show All RPC Handler Tasks Show Active RPC Calls Show Client Operations View as JSON

No tasks currently running on this node.

Software Attributes

Attribute Name	Value	Description
HBase Version	1.0.0, revision=a8fe8a4ebe86d28b93cab3766d2d0f26cd0f7006	HBase version and revision
HBase Compiled	Tue Jul 14 19:18:41 UTC 2015, ec2-user	When HBase version was compiled and by whom
HBase Source Checksum	7210fbdd5a3e3cb4974bd60e9a58bb	HBase source MD5 checksum

HBase Master provides the details about the master node and region servers.

Using HBase Shell

Qubole HBase provides a HBase shell prompt that can be used to query the HBase cluster.

HBase Shell is listed in **Resources** of a running HBase cluster as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)					
		Search : Enter Search Text <input type="text"/> x					
Id		Labels		Nodes	Up Time	Resources	Action
● 4032	✓	hadoop1	default	0		Cluster Usage Report	▶ ✎ ...
● 4033		hadoop2		0		Cluster Usage Report	▶ ✎ ...
● 4034		presto		0		Cluster Usage Report	▶ ✎ ...
● 4035		spark		0		Cluster Usage Report	▶ ✎ ...
● 4131		hbase		2(0)	a few seconds	Cluster Usage Report View Master DNS Hannibal HBase HBase shell HBase Master Ganglia Metrics	✗ ✎ ...

Click **HBase Shell** from the drop-down list and a new shell prompt is displayed.

A **Spark Notebook** is displayed in a new tab/window. Click a Notebook if there is an existing list of notebooks. Click **Create new note** to add a new notebook. The following figure shows a listed note.

The screenshot shows the Qubole Notebook interface. At the top, there is a navigation bar with the Qubole logo, 'Notebook', 'Interpreter', and a 'Connected' status indicator. Below the navigation bar, the main content area displays a heading 'Welcome to Spark Notebook!' and a brief description: 'This web-based notebook enables interactive data analytics. You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!'. Under the heading, there is a section titled 'Notebook' with a 'Create new note' button. Below this, there are two listed notes: 'Note 2AXUAVH3141311441965219623' and 'ExampleNote'.

Click a note and it displays a terminal and type a command on it.

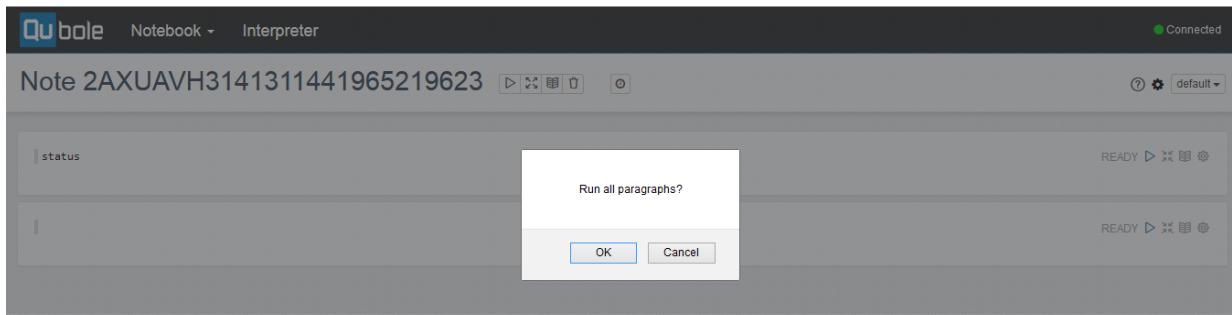
The following figure illustrates a note containing a simple command.

The screenshot shows the Qubole Notebook interface with a specific note selected. The top navigation bar is visible with the Qubole logo, 'Notebook', 'Interpreter', and a 'Connected' status indicator. The main content area shows a terminal window for 'Note 2AXUAVH3141311441965219623'. The terminal window has a header bar with icons for refresh, close, and other controls. The main pane of the terminal shows the text 'status' and a 'READY' status message with a play icon and other icons.

Clicking the run icon ▶ that is next to **Ready** to add another terminal.

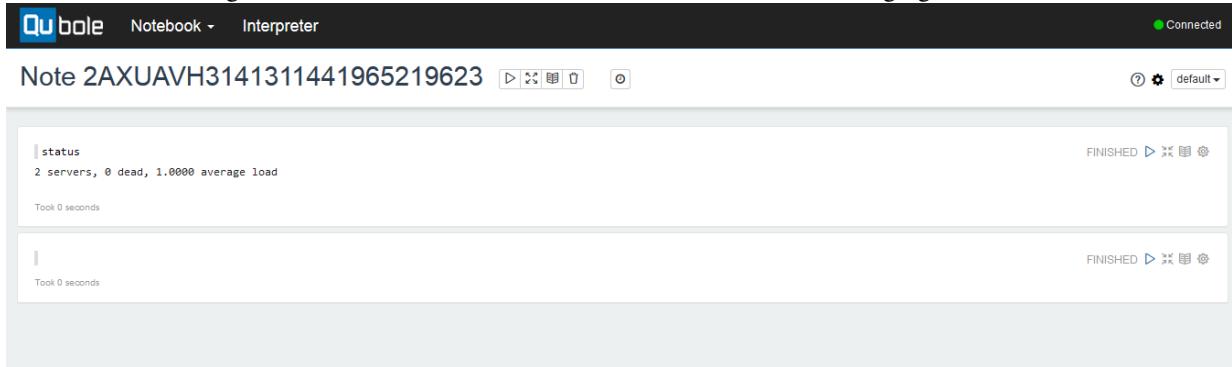
To run the command, click the ▶ that is above the terminal(s) next to the note's name.

A confirmatory dialog is displayed as shown in the following figure.



Click **OK** to run the command. Click **Cancel** if you do not want to run the command.

When the command gets executed, the command result is as shown in the following figure.



2.6.5 Managing an HBase Cluster

You can add any number of slave nodes to an HBase cluster. The size of the HBase clusters is managed by adding/removing the slave nodes that are referred to as region servers. In addition, cluster nodes can be replaced using the HBase cluster management user interface.

To manage an HBase cluster, click the ellipse icon listed in the **Action** column of a running HBase cluster.

The following options are displayed in a drop-down list as shown in the following figure.



Click **Manage** from the drop-down list. **Manage Cluster** is displayed as shown in the following figure.

Cluster > Manage Cluster-4131

Cluster Nodes of My Cluster		+ Add Slave Nodes	
Master Nodes			
Public Dns	Private IP	Up Time	
ec2-54-163-180-194.compute-1.amazonaws.com	ip-10-232-21-156.ec2.internal	4 hours	
Slave Nodes			
Public Dns	Private IP	Up Time	Action
ec2-54-161-166-154.compute-1.amazonaws.com	ip-10-93-136-24.ec2.internal	4 hours	
ec2-54-205-90-139.compute-1.amazonaws.com	ip-10-33-131-49.ec2.internal	4 hours	
ec2-54-144-208-59.compute-1.amazonaws.com	ip-10-232-41-94.ec2.internal	2 hours	

Click **+ Add Nodes** to add nodes to the cluster.

To replace/remove a node, in the Action column of that specific node, pull the downward arrow to see replace and remove options as shown in the following figure.

Cluster > Manage Cluster-4131

Cluster Nodes of My Cluster		+ Add Slave Nodes	
Master Nodes			
Public Dns	Private IP	Up Time	
ec2-54-163-180-194.compute-1.amazonaws.com	ip-10-232-21-156.ec2.internal	2 hours	
Slave Nodes			
Public Dns	Private IP	Up Time	Action
ec2-54-161-166-154.compute-1.amazonaws.com	ip-10-93-136-24.ec2.internal	2 hours	
ec2-54-205-90-139.compute-1.amazonaws.com	ip-10-33-131-49.ec2.internal	2 hours	Remove Node Replace Node
ec2-54-144-208-59.compute-1.amazonaws.com	ip-10-232-41-94.ec2.internal	a minute	

Click **Replace** or **Remove** as required.

Locating Logs on S3

On Amazon S3:

The HBase logs are located at: DEFLOC/logs/hbase.

Where:

- DEFLOC refers to the default location of an account.

2.6.6 Creating HBase Backups and Schedules

A snapshot encapsulates metadata and data in the cluster at a given point in time. It lets an administrator to get back to a previous state of the cluster. An HBase snapshot lets you to take a snapshot of table data with a minimum impact

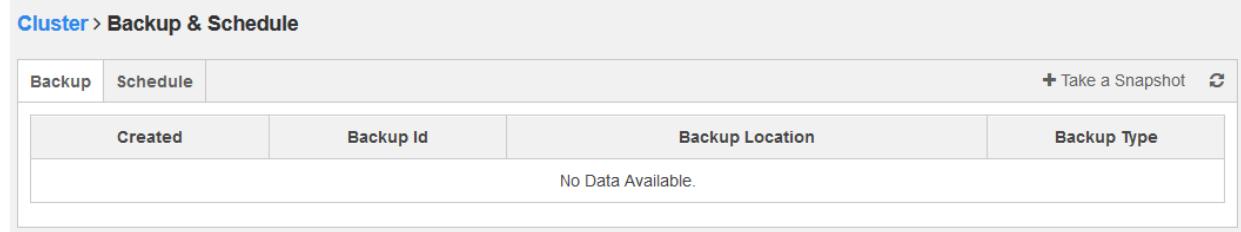
on the Region Servers (slave nodes). [Take an HBase Snapshot](#) provides information on how to create a snapshot using a REST API.

To take a snapshot, click the ellipse icon  listed in the **Action** column of a running HBase cluster.

The following options are displayed in a drop-down list as shown in the following figure.



Click **Backup** from the drop-down list. **Backup and Schedule** is displayed as shown in the following figure.



Click **+ Take a Snapshot** to create a new snapshot. The **Take Snapshot** dialog is displayed as shown in the following figure.

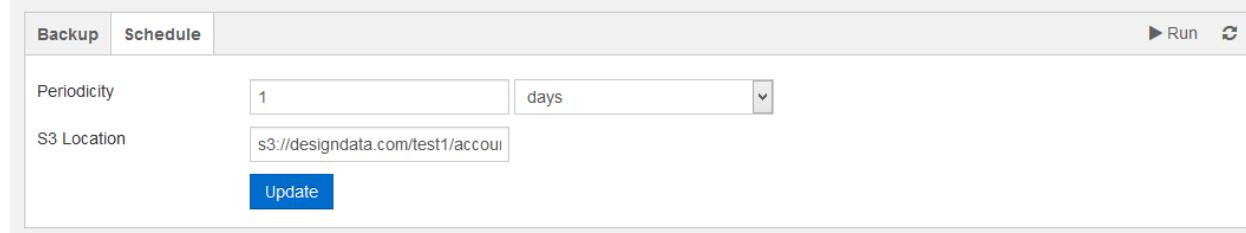


Set the S3 location and set the backup type as **Incremental** if you do not want the default **Full** backup type.

Click **Take Snapshot** to create a snapshot. Click **Cancel** if you do not want to create it.

Click the **Schedule** tab to create a new schedule. **Schedule** is displayed as shown in the following figure.

[Cluster > Backup & Schedule](#)



In **Periodicity** enter the number and select the frequency type, **minutes**, **hours**, **weeks**, or **months** if you do not want the default frequency type, **days**.

Select an S3 location to store the snapshot. Click **Update**. Click the **Run** button to start the schedule.

2.7 Pig in Qubole

2.7.1 Pig in Qubole

Qubole Pig distribution is derived from the Apache Pig version 0.8.

Pig is a platform used to analyze large data sets that contains high-level language to express data analysis programs. Pig's infrastructure layer contains a compiler that generates Map-Reduce programs' sequences, for which large-scale parallel implementations are already existing. Pig's language layer contains a textual language called Pig Latin that has the following important properties:

- **Ease of programming** - Complex tasks containing related data transformations are explicitly encoded as data flow sequences that make them easy to write, understand, and maintain.
- **Optimization Opportunities** - The mechanism in which tasks are encoded lets the system in optimizing the execution automatically and lets you to focus on semantics rather than efficiency.
- **Extensibility** - Allows creating own functions to do special-purpose processing.

Pig version 0.8 (Pig8) is installed in QDS by default. [Running a Pig Job](#) is a quick-start guide to run a pig job query. [Submit a Pig Command](#) provides the REST API information.

Qubole now supports HCatalog and Pig integration. However, only Pig11 and later version supports HCatalog integration. See [Pig HCatalog Integration](#) for more information.

2.7.2 Pig HCatalog Integration

Pig scripts can access Hive metastore using HCatalog libraries. However, only Pig 0.11.1 (Pig11) or later version supports HCatalog integration. As the default version of Pig in Qubole does not support the HCatalog integration, you must install Pig11 version.

Install the Pig11 version by using the following command in the node_bootstrap file.

```
/usr/lib/pig/usr-bin/pig11_install
```

Once the Pig11 version is installed on all the nodes, you can use it from a Shell command UI.

```
/usr/lib/pig/usr-bin/pig11 s3n://bucket/dir/script.pig
```

A Pig code for loading data from a Hive table and storing data into a Hive table is shown in the following example.

```
A = LOAD 'default_qubole_memetracker' USING org.apache.hcatalog.pig.HCatLoader();
B = FILTER A BY month == '2009-04';
STORE B INTO 'default_qubole_memetracker_snipped' USING org.apache.hcatalog.pig.HCatStorer();
```

All data definition language (DDL) statements must be run using the Hive command interface.

2.8 Presto in Qubole

2.8.1 What and Why?

Presto is an open source distributed SQL query engine, developed by Facebook. Presto is used for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes.

Presto was designed and written from the ground up for interactive analytics and approaches the speed of commercial data warehouses. Facebook uses Presto for interactive queries against several internal data stores, including their 300PB data warehouse. Over 1,000 Facebook employees use Presto everyday to run more than 30,000 queries that in total scan over a petabyte each per day. Learn more at prestodb.io

The execution model of Presto is fundamentally different from Hive or MapReduce. Hive translates queries into multiple stages of MapReduce tasks that execute one after the other. Each task reads inputs from disk and writes intermediate output back to disk. In contrast, the Presto engine does not use MapReduce. It employs a custom query and execution engine with operators designed to support SQL semantics. In addition to improved scheduling, all processing is in memory and pipelined across the network between stages. This avoids unnecessary I/O and associated latency overhead. The pipelined execution model runs multiple stages at once, and streams data from one stage to the next as it becomes available. This significantly reduces end-to-end latency for many types of queries. Learn more about Presto's architecture [here](#).

Use Case

Qubole's Presto-as-a-Service is primarily targeted at Data Analysts who are tasked with translating ad-hoc business questions into SQL queries and getting results. Since the questions are often ad-hoc, there is some trial and error involved. Therefore, arriving at the final results may involve a series of SQL queries. By reducing the response time of these queries, the platform can reduce the time to insight and greatly benefit the business.

The typical use-case here is a few tables, each of which is a 10GB-100TB in size, living in Amazon S3. Tables are generally partitioned by date and/or by other attributes. Analyst queries pick a few partitions at a time, typically last one week or one month of data, and involve where clauses. Queries may involve a join with a smaller dimension table and contain aggregates and group-by clauses.

2.8.2 Presto as a Service

Qubole provides Presto as a service for fast, inexpensive and scalable data processing. You can now launch presto clusters with a single-click from the Qubole Data Platform.



Data-formats supported

1. Your hive tables in S3 and HDFS are readily available for querying via Presto.
2. Delimited, CSV, RCFFile, JSON, SequenceFile and ORC file formats are supported.
3. Data compressed using Snappy and LZO codecs is supported.

Cloud Ready

1. Optimize your Presto clusters with your choice of AWS Instance Type.
2. EC2 Classic and EC2 VPC accounts can launch Presto clusters in any AWS region.
3. S3 specific optimizations
4. Save on compute costs with auto-terminated clusters.
5. Auto-scaling Presto clusters adjust to workload
6. Caching to SSDs for interactive performance

Enhanced User Experience

1. Multiple Qubole users can submit queries to the same Presto cluster.

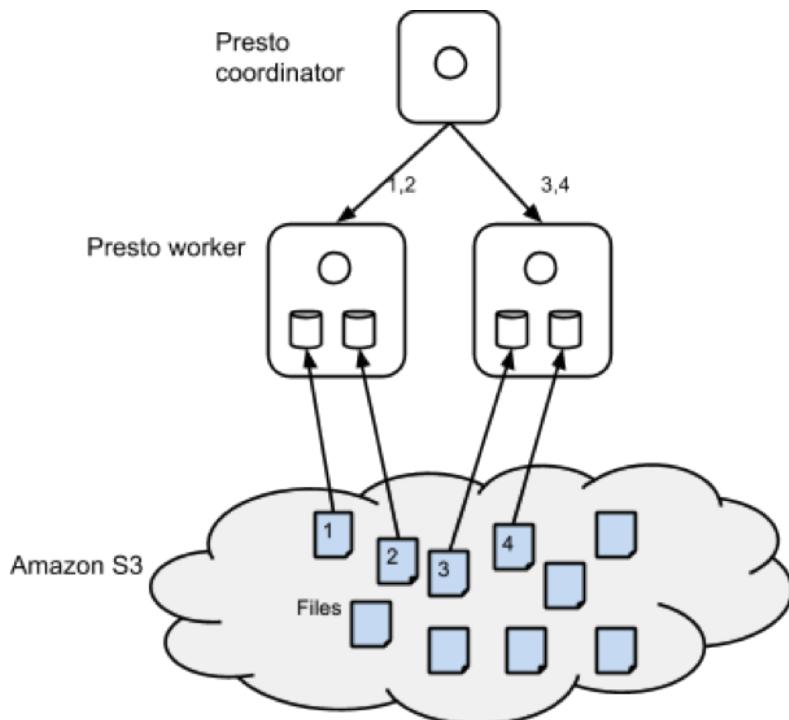
2. Query logs and results are available from the history pane at all times.
3. Qubole provides detailed execution metrics of every Presto query.
4. Mix workflows that involve hadoop jobs, hive queries, and presto queries.
5. Users can configure the amount of memory allocated for Presto in their cluster(s).
6. Launch clusters only when required - when query is run and no cluster is running.
7. Qubole will reuse launched clusters when available for any new queries submitted.

2.8.3 SSD-Caching

Cloud storage (specifically, Amazon S3) has many positives – it is great for storing large amount of data at low cost. However, S3's bandwidth is not high enough to support interactive querying. The new generation of Amazon instance types come with SSD volumes. Some machine types also come with large amount of memory (r3 instance types) per node. We've built a caching framework in Presto to take advantage of this memory hierarchy to provide interactive query performance over large amounts of data in S3.

Architecture

The figure below shows the architecture of the caching solution. As part of query execution, Presto, like Hadoop, performs split computation. Every worker node is assigned one or more splits. For the sake of exposition, we can assume that one split is one file. Presto's scheduler assigns splits to worker nodes randomly. We modified the scheduling mechanism to assign split to a node based on a hash of the filename. This assures us that if the same file were to be read for another query, that split will be executed in the same node. This gave us spatial locality. Then, we modified the S3 filesystem code to cache files in local disks as part of query execution. In the example below, if another query required reading file 2, it will be read by worker node 1 from local disk instead of S3 which will be a whole lot faster. The cache, like all others, contains logic for eviction and expiry. Some instance types contain multiple SSD volumes and we stripe data across them.



We use consistent hashing to handle dynamic addition or removal of nodes due to auto-scaling. This ensures that we maintain the advantage of SSD cache already built in old news as much as possible.

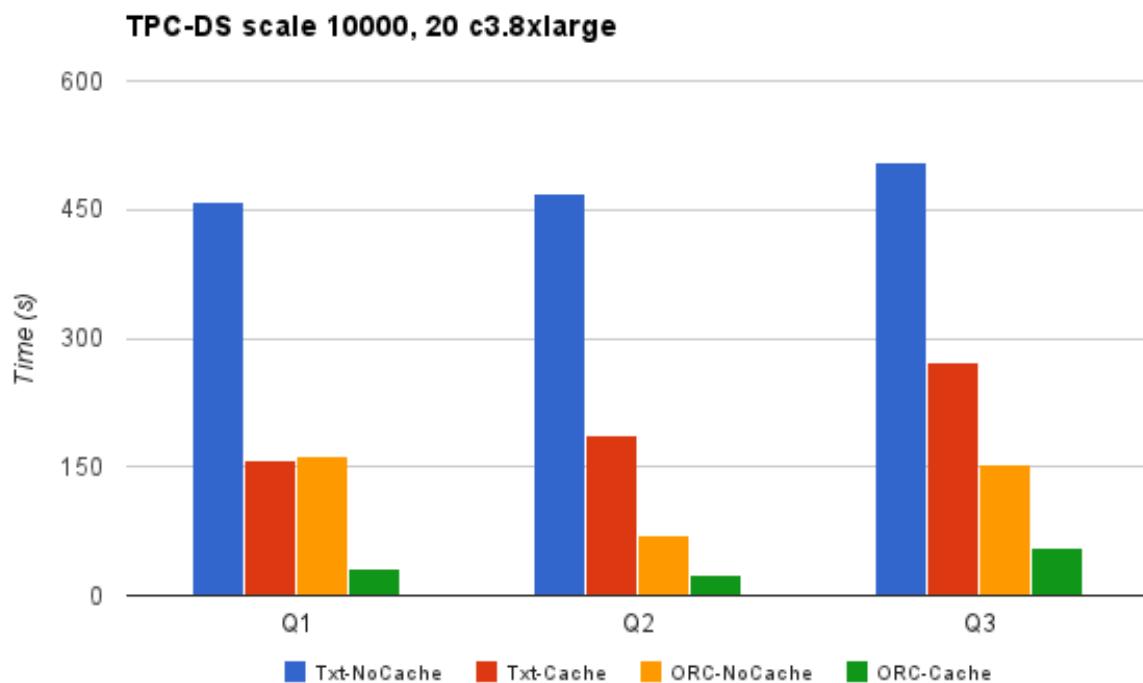
Experimental Results

To test this feature, we generated a TPC-DS scale 10000 data on a 20 c3.8xlarge node cluster. We used delimited/zlib and ORC/zlib formats. The ORC version was not sorted. Here are table statistics.

Table	Rows	Text	Text, zlib	ORC, zlib
store_sales	28 billion	3.6TB	1.4TB	1.1TB
customer	65 million	12 GB	3.1 GB	2.5 GB

We used the following queries to measure performance improvements. These queries are representative of common query patterns from analysts.

ID	Query	Description
Q1	select * from store_sales where ss_customer_sk=1000;	Selects ~400 rows
Q2	select ss_store_sk, sum(ss_quantity) as cnt from store_sales group by ss_store_sk order by cnt desc limit 5;	Top 5 stores by sales
Q3	select sum(ss_quantity) as cnt from store_sales ss join customer c on (ss.ss_customer_sk = c.c_customer_sk) where c.c_birth_year < 1980;	Quantity sold to customers born before 1980



Txt-NoCache means using Txt format with caching feature disabled. The Txt-NoCache case suffers from both problems – inefficient storage format and slow access. Switching to caching provides a good performance improvement. However, the biggest gains are realized when caching is used in conjunction with the ORC format. There is a 10-15x performance improvement by switching to ORC and using Qubole's caching feature. Results show that queries that take many minutes now take a few seconds, thus benefiting the analyst use-case.

Please refer to [Custom Configuration](#) section for details on how to configure SSD caching in your cluster effectively.

2.8.4 Cluster Configuration

A single Qubole account may be associated with multiple clusters. By default, Qubole provides a Presto Cluster along with Hadoop, Hadoop-2, and Spark clusters to each account.

Use the Presto cluster and edit its configuration as required.

To view/edit a Presto cluster's configuration, navigate to **Control Panel > Clusters** and select the cluster with the label **presto**.

The clusters are displayed as shown in the following figure.

Active Cluster(s)		Deleted Cluster(s)			
Search : <input type="text" value="Enter Search Text"/> X					
		Labels	Nodes	Up Time	Resources
Action					
● 4633 ✓	default	0		Cluster Usage Report	
● 4634	presto	0		Cluster Usage Report	
● 4635	hadoop2	0		Cluster Usage Report	
● 4636	spark	0		Cluster Usage Report	

Click the edit icon available in the **Action** column for the *Presto* cluster.

Note: Qubole recommends using `r3.2xlarge` for slave nodes and `r3.xlarge` for a master node.

Cluster settings are displayed as shown in the following figure.

Cluster > Edit Cluster (6921)

Cluster Information									
Cluster Settings									
Cluster Labels	Presto ?								
Cluster Type	Presto ▼								
Minimum Slave Count	1 ?								
Maximum Slave Count	1 ?								
Master Node Type	m1.xlarge - 4 cores, 15GiB m ▼								
Slave Node Type	m1.xlarge - 4 cores, 15GiB m ▼								
Node Bootstrap File	s3://data.com/logs/production/scripts/hadoop/ node_bootstrap.sh								
Other Settings	<input type="checkbox"/> Disable Automatic Cluster Termination ? <input type="checkbox"/> Enable Ganglia Monitoring ?								
Cluster Composition									
Autoscaling Node Purchasing Option	Spot Instance ▼ ?								
Use Qubole Placement Policy	<input checked="" type="checkbox"/> ?								
Volatile Spot Instance Settings									
Fallback to on demand	<input checked="" type="checkbox"/> ?								
Maximum Bid Price	100 % (Pricing: m1.xlarge :\$0.350/hour) EC2 Pricing								
Request Timeout	10 minutes								
Spot Instances Percentage	50 %								
Stable Spot Instance Settings									
Use Stable Spot Nodes	<input type="checkbox"/> ?								
EC2 Settings VALID CREDENTIALS (Sample IAM Credentials)									
Compute Access Key	ABCDEFGHIJKLMNPQRST								
Compute Secret Key	***** Edit Secret Key								
Region	us-east-1 ▼ ?								
Aws Availability Zone	No Preference ▼ ? ↻								
VPC	None ▼ (optional) ↻								
Subnet	▼ (optional)								
Custom EC2 Tags	<table border="1"> <thead> <tr> <th>Custom Tag</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Qubole</td> <td>qbol_acc1</td> </tr> <tr> <td> </td> <td> </td> </tr> <tr> <td colspan="2">+ Add EC2 Tags</td> </tr> </tbody> </table>	Custom Tag	Value	Qubole	qbol_acc1			+ Add EC2 Tags	
Custom Tag	Value								
Qubole	qbol_acc1								
+ Add EC2 Tags									
Security Settings									
Qubole Public Key	ssh-rsa AAAAB3								
Customer Public ssh Key	ssh-rsa AAAAB3 (optional) ?								
Persistent security groups	default (optional) ?								
Enable encryption	<input type="checkbox"/> ?								
2.8. Presto in Qubole									
Override Presto Configuration	?								

Custom Configuration

In cluster settings, there is a textbox for **Override Presto Configuration** which can be used to customize a Presto cluster. An entry in this box can have multiple sections, each section with section title as the relative path of the configuration file in *etc* directory of presto followed by the configuration. You can configure jvm settings, common Presto settings and connectors' settings from here. You can learn more about these sections in [Presto's official documentation](#). Here is an example custom configuration:

```
jvm.config:  
-Xmx10g  
  
config.properties:  
ascm.enabled=false  
  
catalog/hive.properties:  
hadoop.cache.data.enabled=false
```

Some important parameters for each configuration are covered in the following sections.

jvm.config

These are populated automatically and generally do not require custom values. These are used while launching Presto server JVM.

Parameter	Example	Meaning
-Xmx	-Xmx10g	10gb for JVM heap

config.properties

Parameter	Examples	Default	Meaning
ascm.enabled	true, false	true	Enable auto-scaling
ascm.upscaling.enabled	true, false	true	Enable up-scaling
ascm.downscaling.enabled	true, false	false	Enable down-scaling
query.schedule-split-batch-size	1000, 10000	1000	How many splits to schedule at once?
task.max-memory	10g, 20g	80% phymem	Max memory allowed for a task
task.shard.max-threads	10, 20	4 * cores	How many worker threads per JVM

catalog/hive.properties

Parameter	Examples	Default	Meaning
hive.metastore-timeout	3m, 1h	3m	Timeout for hive metastore calls
hive.metastore-cache-ttl	5m, 20m	20m	How long to cache table information
hadoop.cache.data.enabled	false	false	Enable SSD cache
hadoop.cache.data.maxAge	1d	3650d	Cache files younger than interval
hadoop.cache.data.expireInterval	1d	3650d	Expire cached file after interval
hadoop.cache.data.fullPercentage	90s90	95	How much % of SSD disk to fill up
hadoop.cache.data.tableWhitelist	*		Regex whitelisting tables to cache
hadoop.cache.data.tableColumns.min	3	3	Min query columns to cache file
hadoop.cache.fileinfo.enabled	false	true	Cache file size, timestamp
hadoop.cache.fileinfo.expiration	5m	5m	Expire fileinfo cache entry
hadoop.cache.dirinfo.enabled	false	true	Cache directory listing
hadoop.cache.dirinfo.expiration	5m	5m	Expire dir listing from cache
hadoop.cache.data.dirPrefix.list	/media/ephemeral, /media/ephemeralFinal paths created by appending suffix in range [0, 25]	/media/ephemeral, /media/ebs	Prefixes for paths of directories used to store cached data.

Avoiding Stale Caches

The cache parameters are useful to tweak if you expect data to change rapidly.

Example, if a Hive table adds a new partition, it takes Presto 20 minutes to discover it. If you plan on changing existing files in S3, you may want to make *fileinfo* expiration more aggressive. Finally, if you expect new files to land in a partition rapidly, you may want to reduce or disable the *dirinfo* cache.

2.8.5 Connecting to a Presto Cluster Using JDBC

Qubole's JDBC drivers can authenticate a QDS user on a Presto cluster with an API token. See [Managing My Accounts](#) and [Authentication](#) for more information on API tokens.

Advantages of using Qubole JDBC Drivers

Qubole's JDBC drivers provide the following benefits:

- JDBC drivers encrypt data using Secure Sockets Layer (SSL).
- You get a static JDBC endpoint for a cluster. This implies that you can run the same JDBC client program even after the cluster restarts and the master's hostname and IP address change.

Understanding the Prerequisites before using JDBC Drivers

These are the prerequisites required before using Qubole's JDBC drivers:

1. The authentication provided in JDBC drivers is through Qubole. If the master hostname/IP address is exposed to all users, then any user can connect to the Presto cluster through open source Presto JDBC drivers without authentication. So, it is recommended to hide the cluster information from non-admin users.
2. Start the Presto cluster before establishing the connection with JDBC drivers as the cluster must be running for the connection to be established.

3. JDBC drivers cannot prevent clusters from going down or being terminated. So, it is recommended to use the **Disable Automatic Cluster Termination** option in cluster settings. See [Modifying Cluster Settings](#) for more information.

Locating the JDBC Drivers

Download the JDBC drivers from `s3://paid-qubole/presto-jdbc/` and add them to the class path of a Java application.

Configuring a JDBC Driver

Perform the following steps to configure a JDBC driver:

1. Use JDBC URL as: `jdbc:presto://api.qubole.com/<CLUSTER_ID>/<Catalog>/<Database>`.
Example: `jdbc:presto://api.qubole.com/1234/hive/default`
2. Set password as the authentication token.

Sample JDBC Client Program

Here is a sample JDBC client program.

```
import java.net.URI;
import java.net.URISyntaxException;
import java.sql.*;

public class JdbcClient {
    // JDBC driver name and database URL
    static final String JDBC_DRIVER = "com.facebook.presto.jdbc.PrestoDriver";
    static final String DB_URL = "jdbc:presto://api.qubole.com/13098/hive/default";

    // Database credentials
    static final String USER = "ec2-user";
    static final String PASS = "QUBOLE_API_TOKEN_HERE";

    public static void main(String[] args) throws URISyntaxException {
        Connection conn = null;
        Statement stmt = null;
        try{
            //STEP 2: Register JDBC driver
            Class.forName(JDBC_DRIVER);

            //STEP 3: Open a connection
            conn = DriverManager.getConnection(DB_URL,USER,PASS);

            //STEP 4: Execute a query
            stmt = conn.createStatement();
            String sql;
            sql = "select * from nation";
            ResultSet rs = stmt.executeQuery(sql);

            //STEP 5: Extract data from result set
            while(rs.next()){
                //Retrieve by column name
                int nationkey = rs.getInt("n_nationkey");
                ...
            }
        } catch(Exception e){
            e.printStackTrace();
        }
    }
}
```

```

        String name = rs.getString("n_name");
        int regionkey = rs.getInt("n_regionkey");
        String comment = rs.getString("n_comment");

        //Display values
        System.out.println(String.format("%d, %s, %d, %s", nationkey, name, regionkey, comment));
    }
    //STEP 6: Clean-up environment
    rs.close();
    stmt.close();
    conn.close();
}catch(SQLException se){
    //Handle errors for JDBC
    se.printStackTrace();
}catch(Exception e){
    //Handle errors for Class.forName
    e.printStackTrace();
}finally{
    //finally block used to close resources
    try{
        if(stmt!=null)
            stmt.close();
    }catch(SQLException se2){
    }
    try{
        if(conn!=null)
            conn.close();
    }catch(SQLException se){
        se.printStackTrace();
    }
}
System.out.println("Goodbye!");
}
}

```

2.8.6 Your First Presto Query

Ensure that you have followed the steps outlined in [Cluster Configuration](#). You should have created a cluster named *presto*. Go to the **Analyze** page and click the **Compose** button. Select **Presto Query** from the drop-down list. You can run a query against a pre-populated table that will give you the number of flight itineraries in every quarter of 2007. Example:

```
select quarter, count(*) from default_qubole_airline_origin_destination where year='2007' group by quarter;
```

If the cluster is not active, this query will automatically launch it. This can take approximately 3 minutes. Subsequent queries will latch onto the running cluster. You can view the logs indicating progress of the query. Once the query completes you can preview the results. Congratulations! You have just run your first Presto query in Qubole!

2.8.7 Inserting Data

Occasionally, you may wish to write results of a query into another hive table or into an S3 location. Qubole's Presto-as-a-Service includes support for inserting data into (and overwriting) hive tables and S3 directories. We wish to emphasize that Hive is a better option suited for large scale ETL workloads when writing terabytes of data. Presto's insertion capabilities are better suited for 10s of GBs. The insert syntax is very similar to Hive. The insert functionality is currently only available in Qubole's Presto. We are in the process of contributing this back to open source.

We will use the `default_qubole_airline_origin_destination` as the source table in our examples. This table contains flight itinerary information and contains a number of columns and should be available to you.

S3 Directories

You can write the result of a query directly to S3 in delimited format.

```
INSERT INTO directory 's3://qubole.com-siva/experiments/quarterly_breakdown'
SELECT origin,
       quarter,
       count(*) AS c
FROM default_qubole_airline_origin_destination
WHERE YEAR='2007'
GROUP BY quarter,
         origin;
```

Here's a preview of what the result file will look like using `cat -v`. Fields in the results will be ^A (ASCII code \x01) separated.

```
"DFW"^A1^A334973
"LAX"^A1^A216789
"OXR"^A1^A456
"HNL"^A1^A78271
"IAD"^A1^A115924
"ALB"^A1^A20779
"ORD"^A1^A414078
```

Simple Hive Tables

The target hive table can be delimited, CSV, ORC, RCFFile. Inserting into hive tables using custom inputformats and serdes is not supported. You can create a target table in delimited format using this DDL in **Hive**:

```
CREATE TABLE quarter_origin (quarter string, origin string, count int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

Run `desc quarter_origin` to confirm that the table is known to presto. It can take up to 2 mins for Presto to pick up a newly created table in Hive. Now run the following insert statement as a **Presto** query:

```
INSERT INTO TABLE quarter_origin
SELECT quarter,
       origin,
       count(*)
FROM default_qubole_airline_origin_destination
WHERE YEAR='2007'
GROUP BY quarter,
         origin;
```

You can now run queries against `quarter_origin` to confirm that the data has actually landed in the table:

```
SELECT *
FROM quarter_origin LIMIT 5;
```

Similarly, you can overwrite data in the target table by using this query:

```
INSERT overwrite TABLE quarter_origin
SELECT quarter,
       origin,
       count(*)
FROM default_qubole_airline_origin_destination
WHERE YEAR='2007'
GROUP BY quarter,
         origin;
```

Partitioned Hive Tables

The target hive table can also be partitioned. Here's an example target table that is partitioned (run this in **Hive**):

```
CREATE TABLE quarter_origin_p (origin string, count int)
PARTITIONED BY (quarter string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
STORED AS TEXTFILE;
```

Now you can insert into this partitioned table in a similar manner. The only catch is that the partitioning column must appear at the very end of the select list. In the example below the column `quarter` is the partitioning column:

```
INSERT INTO TABLE quarter_origin_p
SELECT origin,
       count(*),
       quarter
FROM default_qubole_airline_origin_destination
WHERE YEAR='2007'
GROUP BY quarter,
         origin;
```

Note that the partitioning attribute can also be a constant. You can also use `overwrite` instead of `into` to erase previous contents of partitions.

2.8.8 Presto Best Practices

ORC Format

Qubole recommends that you use ORC file format with SNAPPY compression along with Presto. It is a fact that ORC outperforms text format considerably. Let us say, you have a table `nation` in delimited form partitioned on column `p`. You can create the ORC version using this DDL as a **Hive** query

```
drop table if exists nation_orc;
create table nation_orc like nation;
alter table nation_orc set fileformat orc;
alter table nation_orc set tblproperties ("orc.compress""=SNAPPY");
SET hive.exec.dynamic.partition = true;
SET hive.exec.dynamic.partition.mode = nonstrict;
insert into table nation_orc partition(p) select * from nation;
```

If `nation` table is not partitioned, then the last 3 lines can be changed to:

```
insert into table nation_orc select * from nation;
```

You can run queries against the newly generated table in Presto and you should see a big difference in performance.

Sorting

ORC format supports skipping reading portions of files if the data is sorted (or mostly sorted) on the filtering columns. As an example, if you expect that your queries are going to filter data on column n_name often, then you can include a sort-by clause while inserting data into the ORC table in **Hive**:

```
insert into table nation_orc partition(p) select * from nation sort by n_name;
```

This layout helps queries like:

```
select count(*) from nation_orc where n_name='AUSTRALIA';
```

Specify Join Ordering

Presto does not do automatic join re-ordering. Hence, make sure that the smaller tables occur on the right side of the JOIN keyword. For example, if table A is larger than table B, then write a join query as follows:

```
SELECT count(*) FROM A JOIN B on (A.a=B.b)
```

A bad join order can slow down a query as the hash table is created on the bigger table. If a bigger table does not fit into the memory, it can cause out of memory exceptions.

Avoiding Stale Caches

The cache parameters are useful to tweak if you expect data to change rapidly. See [catalog/hive.properties](#) for more information.

Example, if a Hive table adds a new partition, it takes Presto 20 minutes to discover it. If you plan on changing existing files in S3, you may want to make *fileinfo* expiration more aggressive. Finally, if you expect new files to land in a partition rapidly, you may want to reduce or disable the *dirinfo* cache.

2.8.9 FAQs

How is Presto different from Hive?

As an user, there are certain differences that you should be aware about Presto and Hive, even though they are able to execute SQL-like queries. **Note:** These differences will change as these technologies evolve and improve. Presto does not:

1. Does not support the IF clause.
2. Does not support User defined functions. However, Presto has a large number of built-in functions.
3. Does not support SUM() as an aggregate-derived window function.
4. Does not support DROP TABLE. Use Hive to drop tables.
5. Does not support CREATE TABLE statement. It does support CREATE TABLE AS SELECT (CTAS). CTAS creates RCFile tables with binary serde. Use Hive to create tables in other formats.
6. Does not support join ordering. User must ensure that smaller table is to the right of the JOIN token.

How is Qubole's Presto different from open-source Presto?

While Qubole's Presto offering is heavily based on open-source Presto, there are a few differences. Qubole's presto:

1. Supports inserting data into hive tables and S3 directories
2. Supports auto-scaling clusters
3. Uses SSDs for caching data from S3 to transparently enhance performance
4. Supports LZO, Zlib compression out of the box
5. Incorporates a number of S3 optimizations that Qubole has successfully used in Hadoop

Where do I locate Presto logs on Amazon S3?

On Amazon S3:

1. The master cluster node's logs are located at: DEFLOC/logs/presto/cluster_inst_id/master/
2. The slave cluster node's logs are located at: DEFLOC/logs/presto/cluster_inst_id/nodeIP/

Where:

- **DEFLOC** refers to the default location of an account.
- **cluster_inst_id** is the cluster instance ID. It is the latest folder in the location, DEFLOC/logs/presto. You can also get it by running a Presto command. When you run a Presto command, the log location are in the **Logs** tab of the QDS user interface (UI), for example, the displayed logs from running a Presto command in the QDS UI **Logs** tab is:

```
Log location: s3://mydata.com/trackdata/logs/logs/presto/95907
Started Query: 20151110_092450_00096_bucas Query Tracker
Query: 20151110_092450_00096_bucas Progress: 0%
Query: 20151110_092450_00096_bucas Progress: 0%
```

In the above sample log location, the Presto cluster instance ID is 95907.

2.9 Spark in Qubole

2.9.1 Introduction

Apache Spark is a fast and general compute engine for Hadoop data. Qubole currently supports Apache Spark **1.3.1**, **1.4.0**, **1.4.1** and beta version of **1.5.0**. The latest supported version is **Spark 1.4.1** and default supported version is **1.4.0**. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation. Spark's in-memory data model and fast processing makes it particularly suitable for applications such as:

- Machine Learning and Graph Processing
- Stream Processing
- Interactive queries against In-Memory data

2.9.2 Cluster Configurations for Spark

By default, each account has a spark cluster configuration and this cluster is used automatically. You can look at the configuration of the **Spark** cluster in **Control Panel**.

Navigate to Control Panel and click the **Clusters** tab in the left pane.

Id	Labels	Nodes	Up Time	Resources	Action
4339	default	0			
4340	presto	0			
4341	hadoop2	0			
4342	spark	0			

Note that by default, there are four cluster definitions, **default**, **hadoop2**, **presto** and **spark**. These clusters are

configured with reasonable defaults. To edit/view a cluster configuration, click the edit icon available in the **Action** column against the **spark** cluster label.

The cluster configuration is as shown in the following figure.

Setting	Value
Cluster Labels	spark
Cluster Type	Spark
Spark Version	1.4.0 (Stable)
Minimum Slave Count	1.3.1
Maximum Slave Count	5
Master Node Type	m3.xlarge - 4 cores, 15GiB m
Slave Node Type	m3.2xlarge - 8 cores, 30GiB r
Ebs Volume Count	0
Ebs Volume Type	Magnetic (standard)
Ebs Volume Size	GB
Node Bootstrap File	s3://data.com/scripts/hadoop/ node_bootstrap.sh
Other Settings	<input type="checkbox"/> Disable Automatic Cluster Termination
	<input type="checkbox"/> Enable Ganglia Monitoring

Specify a label to the cluster to change it and the **Cluster Type** must be Spark. You can select the version from the

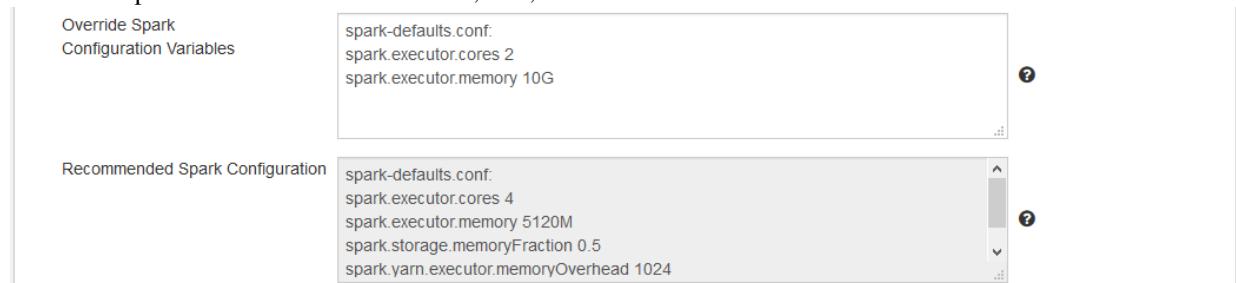
Spark Version drop-down list that contains **1.4.0 (Stable)** selected by default. The supported versions are:

- **1.4.0 (Stable)**
- **1.3.1**
- **1.4.1 (Latest)**
- **1.5.0 (Beta)**

When you change the version while editing a cluster, you must restart the cluster for the changed version settings to be applied to it.

See [Modifying Clusters Configuration](#) for more information on changing other cluster settings.

For spark clusters, Qubole provides the default configuration automatically based on the EC2 slave node type. The corresponding configuration is shown in the above figure. The settings are used by Spark programs running in the cluster irrespective of them run from the UI, API, or SDK.



Note: Use the tooltip to know more information on each field or checkbox.

To change/override the default configuration, provide the configuration values in the **Override Spark Configuration Variables** text box. The format to enter the configuration variables is:

In the first line, enter `-spark-defaults.conf:`. Enter the `<-key value>` in subsequent lines. Provide only one key-value pair per line. An example is as given below.

```
spark-defaults.conf:
spark.executor.cores 2
spark.executor.memory 10G
```

The setting is applied when the cluster is restarted as it does not apply to a running cluster.

You can create multiple Spark clusters in the Control Panel. The [Cluster Operations](#) topic explains how to add a cluster.

Typically, to handle different types of workloads with some being memory-heavy and some being compute-heavy, you can add multiple clusters with each cluster configured differently in Qubole.

2.9.3 Debugging Spark Jobs

The first place to look at debugging Spark jobs is Command logs. By default, Qubole runs Spark jobs on YARN backend and uses yarn-client mode so that driver logs are captured in command logs itself and can be viewed in the [Analyze](#) user interface (UI). See [Composing a Spark Command](#) for more information.

Locating Logs on S3

On Amazon S3:

- The YARN logs (Application Master and container logs) are stored at: `s3://<DEFLOC>/logs/hadoop/CLUSTER_ID/CLUSTER_INST_ID/app-logs`.
- The Spark eventlog files are stored at: `s3://<DEFLOC>/logs/hadoop/CLUSTER_ID/CLUSTER_INST_ID/spark-ev`

Where:

- **DEFLOC** refers to the default location of an account.
- **CLUSTER_INST_ID** is the cluster instance ID. It is the latest folder in the location, `s3://DEFLOC/logs/hadoop/CLUSTER_ID/` for a running cluster or the last-terminated cluster.

Compilation Failures

If you use scala/python code directly in Spark Command, it could fail even before the job is submitted due to compilation issues. Check the logs if YARN application is submitted or not. If you do not see any YARN application related messages, it has failed much before that mostly in compilation phase itself. Usually, the logs contain compilation errors also. (But in some cases for python, compilation failure logs are not captured properly and it will be improved.)

Checking Spark Application UI

Use Spark Application UI to track the progress of the job, memory used by executors/RDDs and so on. This UI captures information if a task/stage fails and it also displays the stack trace.

The container logs can be checked by going to the **Executor** tab in the Spark Application UI. The link to the Spark Application UI is available in the command logs. See [Composing a Spark Command](#) for more information.

Checking Container Logs

Sometimes, it is necessary to check the container logs itself. In YARN, the logs are accessible only after the application finishes. To look at YARN logs for the Spark application, you need the application ID. This ID is displayed in Spark Command logs (for example, `application_1424213537709_0005`). You must SSH into one of the cluster nodes to see container logs. To SSH into cluster nodes, enter the public SSH key in the **Customer Public SSH Key** field by navigating into **Control Panel > Cluster Configuration > Security Settings**. Restart the cluster to SSH into the cluster nodes. **Cluster** page has a column **Nodes**. Clicking it displays the nodes of a cluster. You can SSH into any of the cluster nodes.

After that, issue the following command:

```
/usr/lib/hadoop2/bin/yarn logs -applicationId <applicationId>
```

This command gets the logs of all containers that the application used.

Tuning Configurations

Usually the reasonable way to figure out if the job fails due to resource issues is to give it more resources. In Spark, this is crucial since the resource allocation is static. Try with executor-memory, driver-memory, and num-executors to check if giving more resources actually solves the problem. Later, you can plan to fine tune these configurations to use the resource optimally.

Configuring Spark Job Runtime

Use `yarn.resourcemanager.app.timeout.minutes` to configure the duration in minutes for an YARN application to run. This parameter can prevent runaway applications (if any) from keeping the cluster alive unnecessarily.

It is a cluster-level configuration and must be set in the **Override Hadoop Configuration Variables** available in the **<Cluster Configuration> > Hadoop Cluster Settings** user interface. See [Modifying Hadoop Cluster Settings](#) for more information.

ResourceManager kills an YARN application if it is running for longer than the configured timeout.

ResourceManager never kills an YARN application based on the timeout if it is set to **-1**. On the Spark clusters where you are planning to run Spark streaming jobs, set `yarn.resourcemanager.app.timeout.minutes` to **-1**.

2.9.4 Downgrading to an Older Spark Version

On the Spark clusters that are running version 1.4.0, use the following node bootstraps for downgrading from version 1.4.0 to 1.3.1:

```
hdfs dfs -get s3://paid-qubole/spark/scripts/install_spark_1.3.1_on_rb24 && bash install_spark_1.3.1_on_rb24
hdfs dfs -get s3://paid-qubole/zeppelin/scripts/install_zeppelin_1.3.1_on_rb24 && bash install_zeppelin_1.3.1_on_rb24
```

See [Running Node Bootstrap Scripts on a Cluster](#) for more information.

2.9.5 Creating a Spark Schedule

Click **New Job** in the **Schedule** tab. **Create Job**. See [Schedule](#) for more information on using the using interface to schedule jobs.

Perform the following steps to create a Spark schedule:

1. Enter a name in the **Schedule Name** text field. This field is optional. If it is left blank, a system-generated ID is set as the schedule name.
2. In the **Command** field, select **Spark Command** from the drop-down list. By default, **Scala** is selected as the programming language in the drop-down list that contains Command Line, Python, SQL, and R. Select a language from the list. Enter the query in the text field. The following figure illustrates a Spark Scala query.

Create Job

Schedule Name

Command ?

Spark Command▼

Scala

```
1 import scala.math.random
2 import org.apache.spark._
3 val slices = 6
4 val n = 100000 * slices
5 //spark context is available as sc or spark.
6 val count = sc.parallelize(1 to n, slices).map { i =>
7   val x = random * 2 - 1
8   val y = random * 2 - 1
9   if (x*x + y*y < 1) 1 else 0
10 }.reduce(_ + _)
11 println("Pi is roughly " + 4.0 * count / n)
```

Spark Submit Command Line Options

Spark Submit Default Command line options ?

Arguments for User Program

Cluster Label

Tags ?

Macros ?

3. The **default** label is selected by default in the **Cluster Label** list. Select a different cluster label from the drop-down list.
4. In the **Tags** text field, add one or a maximum of six tags to group commands together. Tags help in identifying commands. Each tag can contain a maximum of 20 characters. It is an optional field.
5. If you have used macros in the query, click the **+Add Macro** button available in the **Macros** field. Else, proceed to the next step. After you click the **+Add Macro** button, the macros are displayed as shown in the following figure.

Macros



The screenshot shows a user interface for defining a macro. At the top, there's a text input field labeled "Variable" containing "Variable" and an equals sign ("="). To its right is another text input field labeled "Expression" containing "Expression". In the bottom left corner of this input area, there's a small "X" icon. Below the input fields are two buttons: a blue "Add Macro" button and a green "Validate Macros" button.

Enter the variable and expression and click **Validate Macros** for validating it. See [Macros in Scheduler](#) for more information. Click **+Add Macro** to add another macro. Else, proceed to the next step.

Note: Use the tooltip  to know more information on each field.

6. **In the Schedule field, set:**
 - (a) **Start Time** by selecting the year, month, date, hour and minute from the corresponding drop-down lists.
 - (b) **End Time** by selecting the year, month, date, hour and minute from the corresponding drop-down lists.
 - (c) **Time Zone** by selecting the appropriate timezone from the drop-down list.
 - (d) **Periodicity** by entering a value and selecting the periodicity type from the drop-down list. **Minutes** is the default periodicity type.
7. Select the number of concurrent jobs allowed from the **Concurrency** drop-down list if you do not want the default value.
8. Select **Skip Missed Instances** if you want to skip instances supposed to have run in the past. By default, this option is unselected. When a new schedule is created, the scheduler runs instances from start time to the current time. For example, if a daily schedule is created from Jan 1 2015 on May 1 2015, jobs are run for Jan 1 2015, Jan 2 2015, and so on. If you do not want the scheduler to run the missed instances for months earlier to May, select the checkbox to skip them.
9. **Dependencies** has three options to be set for a job:
 - **No Dependency** (selected by default)
 - **Wait for Hive Partition**. See [Configuring Hive Tables Data Dependency](#) for more information.
 - **Wait for S3 Files**. See [Configuring S3 Files Data Dependency](#) for more information.
10. **Notification** is an optional field to be selected if you want to be notified through email about instance failure. Once you select the **Send email notifications** checkbox, **Email Type**, **Email List**, and **Event** are displayed as shown in the following figure.

Notification (optional)

Send email notifications

Email Type:

Immediate



Email List:

qubole@qubole.com 

Event:

Failure Success

Select the **Email Type** option, **Daily digest** to receive daily digests if a job periodicity is in minutes or hours. The default email type is **Immediate**.

By default, the current user's email ID is added in the **Email List** field. You can add additional emails as required.

By default, in **Events**, **Failure** and **Success** type of events are not selected. You can select both type of events or any one of them.

11. Click **Submit** to add a new periodic job after you are done with filling the required details. Click **Cancel** if you do not want to submit a job.

See *Schedule* for more information on viewing and editing a job.

2.9.6 Understanding the Spark Job Server

Qubole provides a Spark Job Server that enables sharing of Resilient Distributed Datasets (RDDs) in a Spark application among multiple Spark jobs. This enables use cases where you spin up a Spark application, run a job to load the RDDs, then use those RDDs for low-latency data access across multiple query jobs. For example, you can cache multiple data tables in memory, then run Spark SQL queries against those cached datasets for interactive ad-hoc analysis.

Besides this, you can also use the Job Server to reduce end-to-end latencies of small unrelated Spark jobs. In our tests, we noticed that using the Job Server brought end-to-end latencies of very small Spark jobs down from more than a minute to less than 10 seconds. The major reason for this performance improvement is that in case of the Job Server, you already have a Spark application running to which you submit the SQL query or Scala/Python snippet. On the other hand, without the Job Server, each SQL query or Scala/Python snippet submitted to Qubole's API would start its own application. This happens because the API was designed to run standalone applications.

The following section describes how you can interact with the Spark Job Server using Qubole's [Python SDK](#). Spark Job Server support has been added in SDK version 1.8.0. So, you must update the SDK to that version or to a later version.

Qubole's Spark Job Server is backed by [Apache Zeppelin](#). The main abstraction in the Spark Job Server API is an **app**. It is used to store the configuration for the Spark application. In Zeppelin terms, an app is a combination of a notebook plus an interpreter.

How to use the Spark Job Server

Create a new app using the following command:

```
$ qds.py app create --name app-with-beefy-executors --config spark.executor.memory=20g spark.executor
```

```
{
  "app": {
    "kind": "spark",
    "name": "app-with-beefy-executors",
    "created_at": "2015-10-30T23:42:15Z",
    "updated_at": "2015-10-30T23:42:15Z",
    "qbol_user_id": 1157,
    "deleted_at": null,
    "config": "{\"spark.executor.memory\": \"20g\", \"spark.executor.cores\": \"4\"}",
    "id": 3
  }
}
```

When an app is created, it is in the **DOWN** state and is not associated with any cluster. So, it can be run on any cluster. An app's state changes to **UP** when you submit a command to it and specify a cluster label on which to run the command. As long as the app is in **UP** state, it remains associated with the cluster on which it was started.

You can submit a command to an app by specifying the globally unique app ID (creating an app returns the unique app ID) and the cluster label where the app is running or yet to be run. The following command is an example.

```
$ qds.py sparkcmd run --script_location=some-spark-snippet.py --cluster-label spark --app-id 3
```

When a command is run on an app, the state of the cluster and app can vary as mentioned below:

- The state of the cluster and app can both be **DOWN**. In this case, Qubole starts the cluster and later starts the app on this running cluster, and submits the snippet to the app.
- When the cluster is running and only the app is **DOWN**, Qubole starts the app on this running cluster and submits the snippet to the app.
- When the cluster and app are both **UP**, Qubole submits the snippet to the app.
- When the app is **UP** but on a different cluster, an error message is displayed.

You can continue to submit multiple commands to the app and get results quickly. For example, the following command can also be submitted.

```
$ qds.py sparkcmd run --sql 'select count(*) from default_qubole_memetracker' --cluster-label spark -
```

When you are done with submitting commands, you can mark the app as **DOWN** using the following command:

```
$ qds.py app stop 3
```

The app will get restarted when you submit another command to it.

When a cluster is terminated, all apps associated with it are automatically marked as **DOWN**.

You can list all the apps in an account using the following command:

```
$ qds.py app list

[

  {

    "status": "DOWN",
    "kind": "spark",
    "note_id": null,
    "name": "app-with-beefy-executors",
    "interpreter_id": null,
    "created_at": "2015-10-30T23:42:15Z",
    "qbol_user_id": 1157,
    "cluster_label": null,
    "config": "{\"spark.executor.memory\": \"20g\", \"spark.executor.cores\": \"4\"}"
```

```
        "id": 3
    },
    {
        "status": "UP",
        "kind": "spark",
        "note_id": "2B4S9FQKS1446057961459",
        "name": "concurrent-sql",
        "interpreter_id": "2B5NE7CKT1446057961437",
        "created_at": "2015-10-31T18:45:05Z",
        "qbol_user_id": 1157,
        "cluster_label": "spark",
        "config": "{\"zeppelin.spark.concurrentSQL\":\"true\"}",
        "id": 5
    }
]
```

You can view a particular app using the following command.

```
$ qds.py app show 3

{
    "status": "DOWN",
    "kind": "spark",
    "note_id": null,
    "name": "app-with-beefy-executors",
    "interpreter_id": null,
    "created_at": "2015-10-30T23:42:15Z",
    "qbol_user_id": 1157,
    "cluster_label": null,
    "config": "{\"spark.executor.memory\":\"20g\", \"spark.executor.cores\":4}",
    "id": 3
}
```

Performing ELB Log Analysis

Here is an example, which shows how you can use the Spark Job Server to do ELB log analysis:

1. Create a new app, which allows concurrent execution of SQL queries.

```
$ qds.py app create --name elb-log-analysis-demo --config zeppelin.spark.concurrentSQL=true
```

2. Submit `elb-parsing-definitions.py` script to this app ID.

```
$ qds.py sparkcmd run --script_location=elb-parsing-definitions.py --cluster-label spark --app-i
```

3. Submit the `elb-log-location.py` script to this app ID. This specifies the ELB log location and registers the cached data as a temporary table. You can execute this step multiple times to cache different data locations in memory as different tables.

```
$ qds.py sparkcmd run --script_location=elb-log-location.py --cluster-label spark --app-id 3
```

4. Now that there is data in memory, it can be analyzed by running the following queries:

```
$ qds.py sparkcmd run --sql 'select ssl_protocol, count(*) as cnt from logs group by ssl_protocol'
$ qds.py sparkcmd run --sql 'select * from logs where backend_status_code="500"' --cluster-label
$ qds.py sparkcmd run --sql 'select * from logs order by backend_processing_time desc limit 10'
```

All these queries would return quickly because they use in-memory data.

But it is important to note that you can run any other unrelated query or program and even that would also return quickly because it would execute against an already running app. So, for example, you can run the following command:

```
$ qds.py sparkcmd run --sql 'select count(*) from default_qubole_memetracker' --cluster-label spark -
```

The above query would return quickly as well.

2.10 Qubole Scheduler

See *Schedule* for information on using the Qubole user interface to schedule periodic jobs.

2.10.1 Hive Datasets as Schedule Dependency

This section describes how schedules can be setup to run ONLY if the data is available in Apache Hive tables. Typically, schedules which run Hive commands depend on data in Hive tables. For the purpose of illustration, the following table is used.

```
CREATE EXTERNAL TABLE daily_tick_data (
    date2 string,
    open float,
    close float,
    high float,
    low float,
    volume INT,
    average FLOAT)
PARTITIONED BY (
    stock_exchange STRING,
    stock_symbol STRING,
    year STRING,
    date1 STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION 's3n://paid-qubole/default-datasets/stock_ticker';
```

date1 is a date with format YYYY-MM-DD. The dataset is available from 2012-07-01. For this example, let us assume that the data set is updated everyday at 1AM UTC and jobs are scheduled everyday at 2AM UTC.

The following query has to be executed everyday:

```
SELECT stock_symbol, max(high), min(low), sum(volume) FROM daily_tick_data WHERE date1='`$yesterday`'
```

The following sub-topics are described in this page:

- *Partition Column Values*
- *Dataset Interval*
- *Initial Instance*
- *Understanding S3 Files Dependency*
- *Understanding Hive Partition Dependency*

Partition Column Values

Qubole has to be informed about the new partitions that are added everyday.

In the example, the following partitions are added on 2013-01-02:

stock_exchange = nasdaq	stock_symbol = ibm	year = 2013	date1=2013-01-01
stock_exchange = nasdaq	stock_symbol = orcl	year = 2013	date1=2013-01-01
stock_exchange = nyse	stock_symbol = ibm	year = 2013	date1=2013-01-01
stock_exchange = nyse	stock_symbol = ibm	year = 2013	date1=2013-01-01

For example, the partition columns can have the following values:

stock_exchange	[nasdaq, nyse]
stock_symbol	[ibm, orcl]
year	%Y
date1	%Y-%m-%d

The above information has to be entered while submitting a job either through the UI or API.

The format of the partition columns, year and date1, does not change from one job to another. These are stored in the Hive metastore and does not need to be specified every time.

The format for date partition columns can be entered through the UI or the API. For more information on Store Table Properties, see [Store Table Properties](#).

See [Configuring Hive Tables Data Dependency](#) for more information on setting hive table data dependency using the Qubole user interface.

Dataset Interval

In the previous example, the job runs everyday and the dataset is generated everyday. It is possible that the job runs at a different frequency, than the interval at which the dataset is generated. For example, the following query is run once in seven days while the dataset is generated once a day.

```
SELECT stock_symbol, max(high), min(low), sum(volume) FROM daily_tick_data WHERE date1> '$sevendaysago'
```

Qubole needs additional information to schedule this job as mentioned in the following table.

interval	How often the data is generated.
window_start, window_end	Defines the range of intervals to wait for. Each is an integer which is multiple of the interval.

For this example, the values for interval, window_start and window_end are:

interval	1 day
window_start	-6 (inclusive of seven days ago)
window_end	0 (inclusive of today)

Just like date formats of the partition columns, the interval at which the dataset is generated does not change often. Interval can also be stored in the Hive metastore and need not be specified every time.

See [Configuring S3 Files Data Dependency](#) for more information on setting S3 files data dependency using the Qubole user interface.

Initial Instance

Initial instance specifies the first instance of the data that is available. This is useful when a new dataset is introduced. It is possible that some jobs at the beginning may not have all instances available and should not be generated.

Let us understand the dependency of data in S3 files and Hive partitions required by the Qubole Scheduler for scheduling jobs. Dependencies are the prerequisites that must be met before a job can run.

Understanding S3 Files Dependency

S3 files' dependency implies that a job runs if the data is arrived in S3 buckets. You can schedule a job to run at a specific date and time, either once or on a repetitive basis if the data exists. You can define repeat intervals such as last 6 hours, last 6 days, last 3 weeks, and last 7 months.

To schedule jobs at periodic intervals, Qubole Scheduler requires the following information:

- Start day or time (parameter: window_start)
- End day or time (parameter: window_end)
- Day or time interval that denotes when and how often data is generated (parameter: interval)
- Nominal time which is the logical start time of an instance

The dependency must be defined as: `s3://<bucket>/<folderinS3bucket>/<abc>-%Y-%m-%d-%H-%M-%S`. For example, `s3://abc.com/data/schedule-2014-12-31-00-00-00` is a sample dependency.

See [Time class](#) for more information on date and time placeholders.

The following table shows how to create data in S3 files for the previous day's data with daily interval.

Sequence ID	Nominal Time	Created At	Dependency
1	2015-01-01 00:00:00	2015-04-22 10:00:00	<code>s3://abc.com/data/schedule-2014-12-31-00...</code>
2	2015-01-02 00:00:00	2015-04-22 10:15:00	<code>s3://abc.com/data/schedule-2015-01-01-00...</code>
3	2015-01-03 00:00:00	2015-04-22 10:30:00	<code>s3://abc.com/data/schedule-2015-01-02-00...</code>

The window_start and window_end parameters are relative to Nominal Time.

Interpreting window_start Parameter Values

The value 0 implies now, -1 implies 1 day ago, and -2 implies 2 days ago.

Correspondingly, for hourly/daily/weekly/monthly/yearly interval (frequency), the value 0 denotes now. -1 denotes 1 hour/day/week/month/year ago. -2 denotes 2 hour/day/week/month/year ago and so on.

Interpreting window_end Parameter Values

Qubole Scheduler supports waiting for data. For example, waiting for 6 weeks of data implies that window_start is -5 and window_end is 0.

When the data arrival interval and the scheduler interval are different, then the scheduler interval follows its own frequency to process the data. For example, if the data arrival interval is hourly and the scheduler interval is daily, the scheduler waits for an entire day's data.

Data and the scheduler can be in two different timezones. For example,

```
{
  window_start => -48
  window_end => -24
  frequency => hourly
  time_zone => Americas/Los_Angeles
}
scheduler_frequency => daily
time_zone => Americas/New_York
```

Understanding Hive Partition Dependency

Qubole Scheduler allows data units to have hive partitions. Data in Hive tables can be categorized by Hive partitions such as country or date. The Hive query example to create external table mentioned above contains hive partitions. The scheduler recognizes the Hive partitions from the corresponding table properties available in the Hive metastore. See [Partitions](#) for more information.

Timezone can be specified as an optional parameter in a hive query with daylight savings as on/off.

Hive tables can be partitioned by *date* and *country*. Dependency is expressed as %Y/%M/%d/["US", "CAN", "IRE"].

2.10.2 Macros in Scheduler

In the Qubole Scheduler, commands need access to the context of the instance. The scheduler provides access to the context through macros.

New macros can be defined using the Javascript language. The *daily_tick_data* table is used in the example given below. The example query is:

```
SELECT stock_symbol, max(high), min(low), sum(volume)
FROM daily_tick_data
WHERE date1 > '$sevendaysago$' AND date1 <= '$yesterday$'
GROUP BY stock_symbol
```

Macros can be accessed or defined using the Javascript language. Only assignment statements are valid. Loops, function definitions, and all other language constructs are not supported. Assignment statements can use all operators and functions defined for the objects used in the statements. Defined macros can be used in subsequent statements.

Javascript Language and Modules

The following Javascript libraries are available.

Library	Description	Link to Documentation
Moment.js	Provides many date/time related functions.	Moment.js
Moment-tokens	Provides strftime formats	Moment-tokens

The macros shown in the query are defined as follows:

```
sevendaysago = Qubole_nominal_time.clone().subtract('days', 7).strftime('%Y-%m-%d')
yesterday = Qubole_nominal_time.clone().subtract('days', 1).strftime('%Y-%m-%d')
```

System Variables

The system variables are described in the following table.

Qubole_nominal_time	A moment object representing the time when this instance is supposed to run.
Qubole_nominal_time_iso	Qubole_nominal_time is in ISO 8601 format.

See [Creating a New Job](#) for more information on setting Macros using the Qubole user interface.

2.11 Qubole Billing Guide

2.11.1 Introduction

Qubole rents machines from AWS/GCE/Azure on behalf of a customer and works against data stored in their storage accounts. The customer is responsible for the raw compute/network/storage costs incurred within their accounts and pays that directly to the Cloud vendor. Qubole charges a premium for its service based on usage.

2.11.2 QCUH

The Qubole Compute Unit Hour (QCUH) is a normalized unit of compute used for accounting purposes. It is defined to be equal to the use of one **m1.xlarge** instance for one hour in the us-east region on AWS. We normalize other instance types in different clouds/regions based on latest on-demand rates and machine configurations.

Tables with QCU ratings for different instance types in AWS *⁰ and GCE †⁰ are provided below.

0

AWS Instance Type	Equivalent QCU
t1.micro	0.0417
m1.small	0.125
m1.medium	0.25
m1.large	0.5
m1.xlarge	1
m2.xlarge	0.704545
m2.2xlarge	1.3977
m2.4xlarge	2.7954
m3.medium	0.1989
m3.large	0.3978
m3.xlarge	0.7955
m3.2xlarge	1.5909
m4.large	0.3580
m4.xlarge	0.7159
m4.2xlarge	1.4318
m4.4xlarge	2.8636
m4.10xlarge	3.0682
c1.medium	0.3409
c1.xlarge	1.3636
c3.large	0.3011
c3.xlarge	0.6022
c3.2xlarge	1.1932
c3.4xlarge	2.386
c3.8xlarge	3.0681
c4.large	0.3125
c4.xlarge	0.6250
c4.2xlarge	1.2528
c4.4xlarge	2.5057
c4.8xlarge	3.0682
cc2.4xlarge	2.03125
cc2.8xlarge	3.0681
cg1.4xlarge	3.0681
hi1.4xlarge	3.0681
r3.large	0.5114
r3.xlarge	1.0227
r3.2xlarge	2.0454
r3.4xlarge	3.0681
r3.8xlarge	3.0681
i2.xlarge	2.4204
i2.2xlarge	3.0681
i2.4xlarge	3.0681
i2.8xlarge	3.0681
d2.xlarge	1.9659
d2.2xlarge	3.0681
d2.4xlarge	3.0681
d2.8xlarge	3.0681

GCE instance type	Equivalent QCU
n1-standard-1	0.25
n1-standard-2	0.50
n1-standard-4	1.00
n1-standard-8	2.00
n1-standard-16	4.00
n1-highmem-2	0.59
n1-highmem-4	1.17
n1-highmem-8	2.34
n1-highmem-16	4.69
n1-highcpu-2	0.31
n1-highcpu-4	0.63
n1-highcpu-8	1.26
n1-highcpu-16	2.51

2.11.3 Billing Model

The premium charged by Qubole is based on three units of consumption:

- Number of QCUH used
- Number of Users using Qubole

Our pricing includes an **Elastic** tier (where users are billed based on consumption on a monthly basis) and convenient packages at fixed monthly rates. For more information on price sheets and details, refer to the [Pricing page](#).

[**Amazon Web Services - QCU Mapping**](#)

[**Google Compute Engine - QCU Mapping**](#)

Qubole Administration Guide

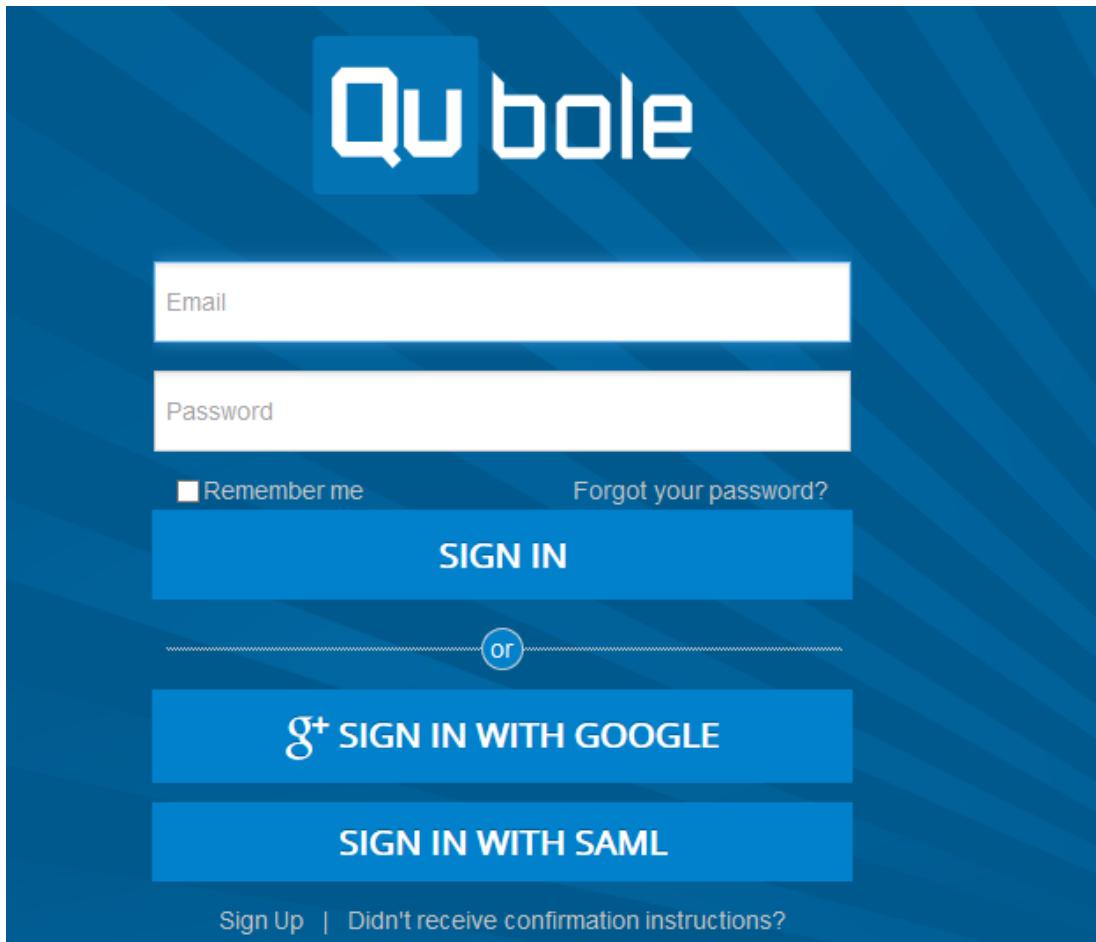
3.1 Introduction

This guide contains information on administering the Qubole Data Service (QDS). It is intended at system administrators and users managing QDS.

3.2 Using SAML Single SignOn and Google Authorization Service

Qubole supports logging in using username and password and through google authorization and SAML Single Sign-On (SSO) after signing up. To use google authorization and SSO for signing into QDS, send an email to help@qubole.com.

The following figure shows the QDS login page.



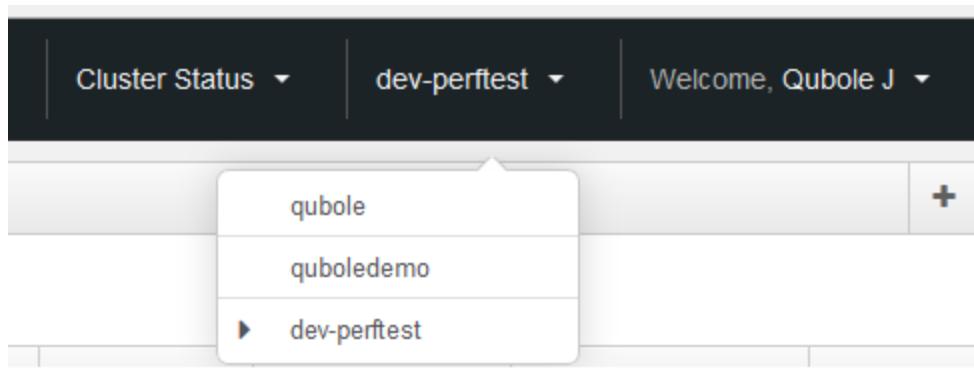
3.3 Managing the Account Settings

[Account Settings](#) displays the configuration of only the account that is in use currently. To view/change any other account's configuration, switch to that account and view/modify its settings.

3.3.1 Switching Accounts

Navigate to the [My Accounts](#) page, click an account to which you want to switch to. See [Managing My Accounts](#) for more information.

Alternatively, click the drop-down list that is against the current account on the top-left corner of the QDS UI as illustrated in the following figure.



The drop-down list displays the list of accounts. Click the account to which you want to switch to. (The arrow pointer is against the current account that is in use).

3.3.2 Modify Account Settings

Managing Account Settings describes how to modify the account's configuration.

3.3.3 Modify Compute Credentials

Managing Account Settings describes how to modify the account's compute credentials.

3.3.4 Modify Storage Credentials

Managing Account Settings describes how to modify the account's storage credentials.

3.3.5 Allowing Domains to Sign Up

Managing Account Settings describes how to set domains to sign in and sign up.

3.4 Managing Access Permissions and Roles

QDS allows system administrators to create custom roles for granting/rejecting access to specific resources. A role has a list of policies that determine which features of QDS can be accessed by a user logged into QDS. See *Manage Roles* for more information.

Types of Roles briefly describes the system-defined roles and custom roles. *Precedence of Roles* briefly describes how assigning multiple roles to a group works.

For adding, modifying, and deleting roles, click **Manage Roles** in the **Control Panel** page.

3.4.1 Create a Role

Creating a Role explains the steps to add a new role.

3.4.2 Modify Access Permissions to a Role

Modifying a Role describes the steps to modify the access permissions to a role.

3.4.3 Delete a Role

You can only delete a custom-role. System-defined roles cannot be deleted. See *Types of Roles* for more information. In **Manage Roles**, click the downward arrow in the **Action** column and select **Delete** to remove a role.

3.4.4 Types of Roles

There are two types of roles:

- **System-defined roles:**
 - System-admin - It is a system-defined role, which has full access to all resources. People, who are part of the group of the same name get full access to all resources in a Qubole account.
 - System-user - It is also a system defined role, whose access is restricted. It allows people, who are part of the group with this role to read all Qubole resources (such as commands and templates), create commands in **Analyze**, create, clone and run Templates, create, clone Schedules, kill and rerun Scheduler Instances. Create, update, and delete operations on data stores in **Explore** are denied. Create, clone, update, and delete cluster operations are also denied.
- **Custom roles:** Apart from these system defined roles, you can create roles with a definition allowing granular control over the features of the Qubole user interface. These are the roles with specific permissions identified by a user-defined name. A system administrator can create custom roles.

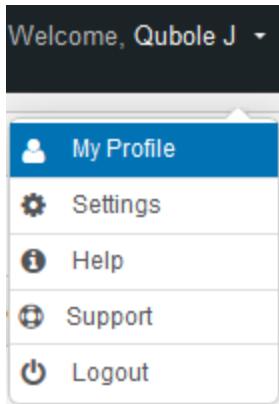
3.4.5 Precedence of Roles

When a single resource with allow and deny restrictions across two roles in the same group is applied, the least restrictive policy applies. For example, if Role1 denies read access to commands and Role2 allows read access to commands. If Role1 and Role2 are part of a group named Group1, then, while applying these rules, Role2's policy gets precedence over Role1's policy in Group1.

3.5 Managing a User Profile

3.5.1 Access a Profile

A user profile can be accessed from the **Control Panel** as well as drop-down list that shows the profile name at the top-right corner of the QDS UI as shown in the following figure.



Alternatively, navigate to the Control Panel and click [My Profile](#).

3.5.2 Modify a Profile

See [Managing Profile](#) on how to change a profile name and password.

3.6 Adding and Managing Users

Qubole lets a user join many accounts with a single profile. A user can have one default account at a given point of time. Hence, an account on QDS can have many users and it can be a part of more than one group.

3.6.1 Viewing Users in an Account

Navigate to the **Control Panel** and click [Manage Users](#) to see the list of users in a specific account.

3.6.2 Inviting a User to Join an Account

See [Managing Users](#) for information on inviting a user to an account. When inviting a new user, you have the option to add the new user to a group. If no group is mentioned, by default, the user is added to the **system user** group.

3.6.3 New User Signing Up to Join an Existing Account

[Manage Users](#), click the hyperlink that is against **New user can join your account while signing up using this link:** and complete the signup form to join the account. That hyperlink can be copied and shared over an email/instant message. A new user, who signs up using that hyperlink automatically joins that particular account.

See [AWS Quick Start Guide](#) for more information on how to fill the signup form.

3.6.4 Activating a Pending User

If you are system administrator, then you see a list of users who have signed up to join that particular account in the **Pending Users** tab of [Manage Users](#). Activate the pending users by approving the requests.

3.7 Creating and Managing Groups

Groups are used to simplify authentication and permissions. Groups and roles allow fine-grained authorization of various parts of QDS. A group is a collection of users and roles.

Navigate to the **Control Panel** tab and click [Manage Groups](#) to see the list of groups to which the account that is being currently used is part of. You can create/modify/delete a group on [Manage Groups](#).

3.7.1 Create a Group

[Managing Groups](#) describes how to create a group. After a group is created, it is visible in the list of groups.

3.7.2 Modify a Group

[Managing Groups](#) describes how to modify a group.

3.7.3 View Users in a Group

In [Manage Groups](#), click the downward arrow in the **Action Column** and click **Show Users** to see the list of users in a group.

3.7.4 Delete a Group

[Managing Groups](#) describes how to delete a group.

3.8 Enabling Encryption for Data at Rest on Amazon S3

For security reasons, data at rest may have to be encrypted. When working with Qubole on Amazon Web Service, data is at rest in these two locations:

1. S3
2. The ephemeral HDFS brought up on the EC2 compute nodes

3.8.1 Enable Encryption on S3

Qubole leverages S3's server-side encryption. For more information, see this reference.

To enable this server-side encryption, set the following property:

```
"fs.s3n.sse=AES256" (default='None')
```

The property can be set in the following ways:

1. As a hadoop configuration override. This is set at the **Add/Edit Cluster** page, **Override Hadoop Configuration Variables**. Here it would be set as **fs.s3n.sse=AES256**. (Navigate to the Control Panel page. In the **Clusters** tab, click the edit icon  to go to the Edit Cluster page or + icon to go the Add Cluster page).
2. As a hive bootstrap setting. This would affect all hive commands for a given account. Here the syntax to use is: set **fs.s3n.sse=AES256**.

3. As a per command setting. For hive commands, you can set it per query using **set fs.s3n.sse=AES256;** in the same command session as the query.

For example,

```
CREATE EXTERNAL TABLE New2 (`Col0` STRING, `Col1` STRING, `Col2` STRING) PARTITIONED BY ( `20100102` ... )
```

Note:

The results of the select calls with the limit clause are not encrypted as the limit clause would result in bypassing of the map/reduce flow.

Results of select calls without the limit clause are encrypted. Basically, a standard Hadoop map/reduce output is encrypted. A Presto output, which does not use map/reduce is not encrypted.

3.8.2 Enable Encryption on Ephemeral HDFS through QDS UI

Navigate to the [Control Panel](#) page. In the **Clusters** tab, click the edit icon  to go to the Edit Cluster page.

Select **Enable Encryption** listed below **Security Settings** as shown in the following figure.

Security Settings

Qubole Public Key	<input type="text" value="ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCyODZaLi2DgacYHgLRCm1Sfwl"/>
Customer Public ssh Key	<input data-bbox="530 1003 1330 1045" type="text"/> (optional)
Enable encryption	<input type="checkbox"/> 
<input type="button" value="Save"/> <input type="button" value="Cancel"/>	

Enable Encryption is an option to encrypt the data at rest on the node's ephemeral (local) storage. This includes HDFS and any intermediate output generated by Hadoop. The block device encryption is setup before the node joins the cluster and can increase the bring up time of the cluster.

3.9 Installing the Qubole ODBC Driver

The Qubole ODBC driver allows you to access data in Qubole Data Service from Business Intelligence applications such as Microsoft Excel and Tableau.

Qubole currently provides only Microsoft © Windows ODBC drivers as listed below:

- QuboleUL_x86.msi for Microsoft Windows 32-bit edition
- QuboleUL_x64.msi for Microsoft Windows 64-bit edition

Write to help@qubole.com to request for Qubole ODBC drivers.

3.9.1 Prerequisites

Qubole ODBC Driver supports 32-bit and 64-bit editions of the following Microsoft Windows versions:

- Microsoft Windows 7

- Microsoft Windows 8

3.9.2 Install the Qubole ODBC Driver

In this example, assume that you want to install ODBC driver on a 32-bit (x86) edition of Microsoft Windows. However, the steps to install ODBC drivers on Microsoft Windows 32-bit and 64-bit edition remain the same.

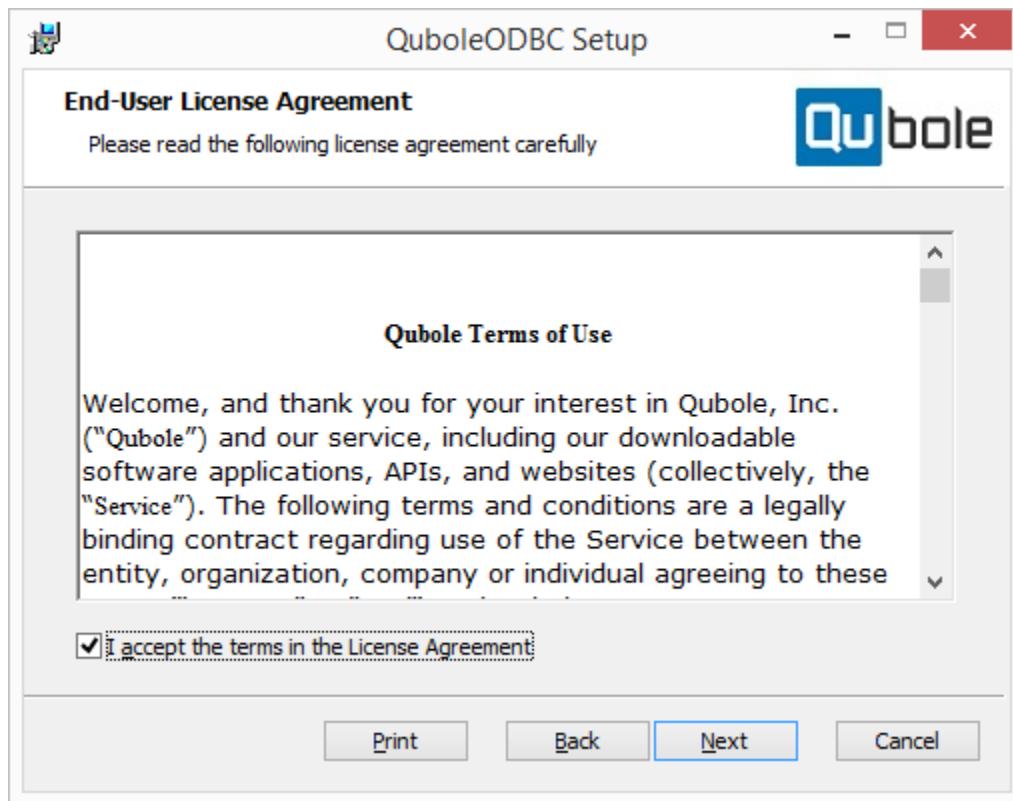
Perform the following steps to install ODBC driver on Microsoft Windows:

1. Click QuboleUL_x86.msi for 32-bit to run it. (Click QuboleUL_x64 for 64-bit.)
2. A welcome screen is displayed as shown in the following figure.



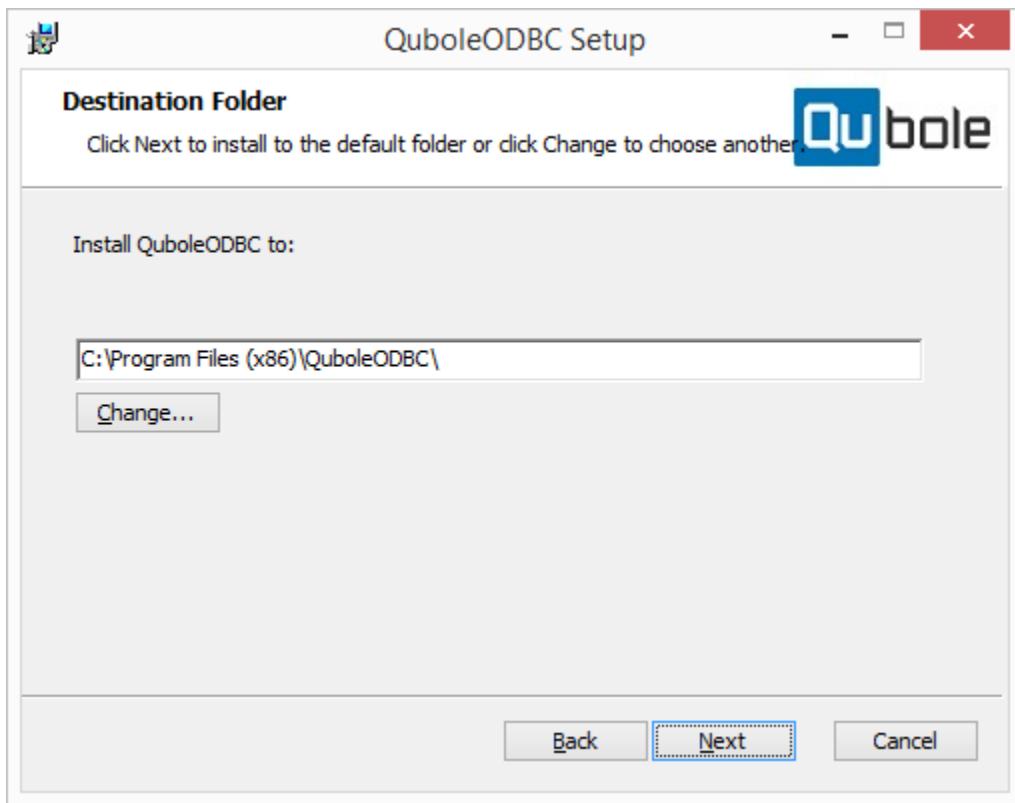
Click **Next** to continue. Click **Cancel** if you want to cancel the installation.

3. Clicking **Next** displays the end-user licence agreement as shown in the following figure.



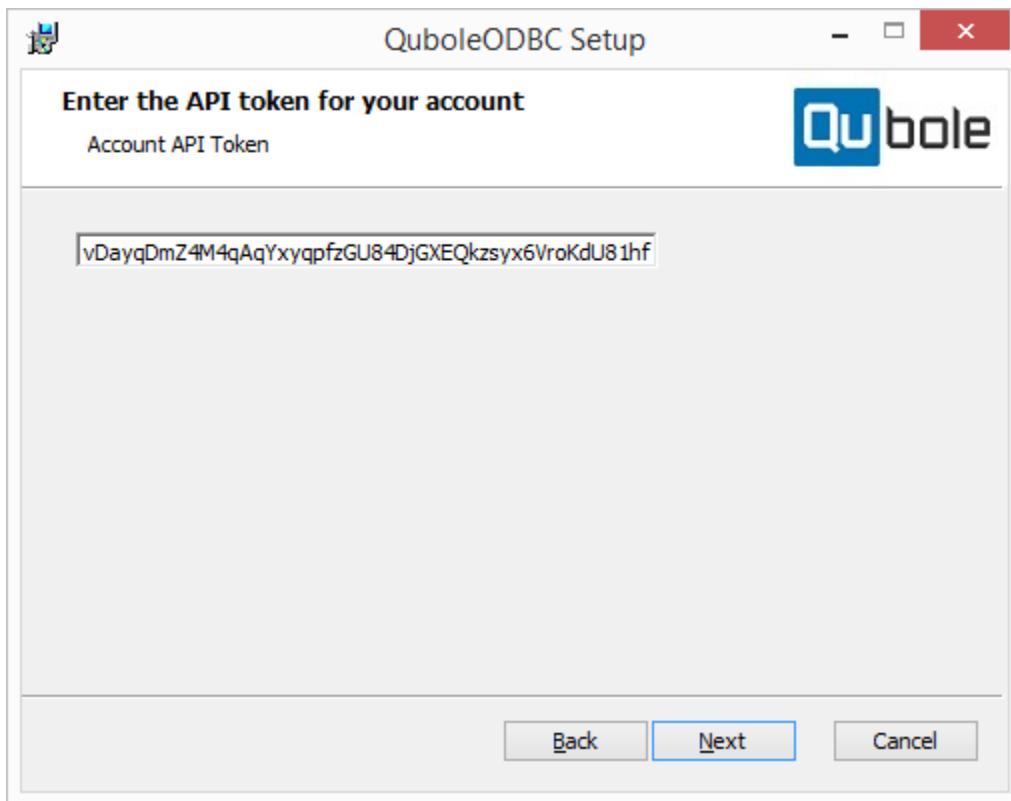
Read the terms and select **I agree to the terms in the Licence Agreement**. Click **Next** to continue only after accepting the terms. Click **Back** to go to the previous dialog and click **Cancel** if you want to cancel the installation.

4. Clicking **Next** displays the dialog to set the destination folder as shown in the following figure.



A default location is as shown in the above figure. If you want a different location, click **Change** and browse to select a required location. Click **Next** to continue. Click **Back** to go to the previous dialog and click **Cancel** if you want to cancel the installation.

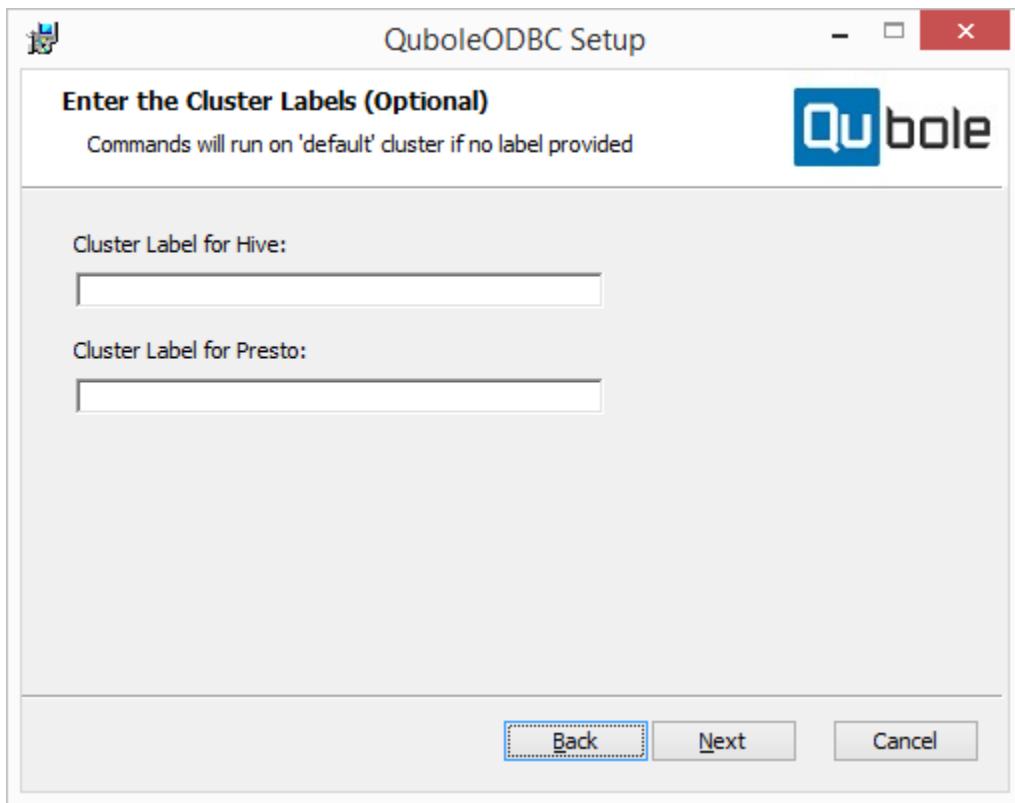
5. Clicking **Next** displays the dialog to set the current account's API token as shown in the following figure.



Enter the API token of your current account. You can see the API tokens in **Control Panel > My Accounts**. See [Managing My Accounts](#) for more information on accounts and API tokens.

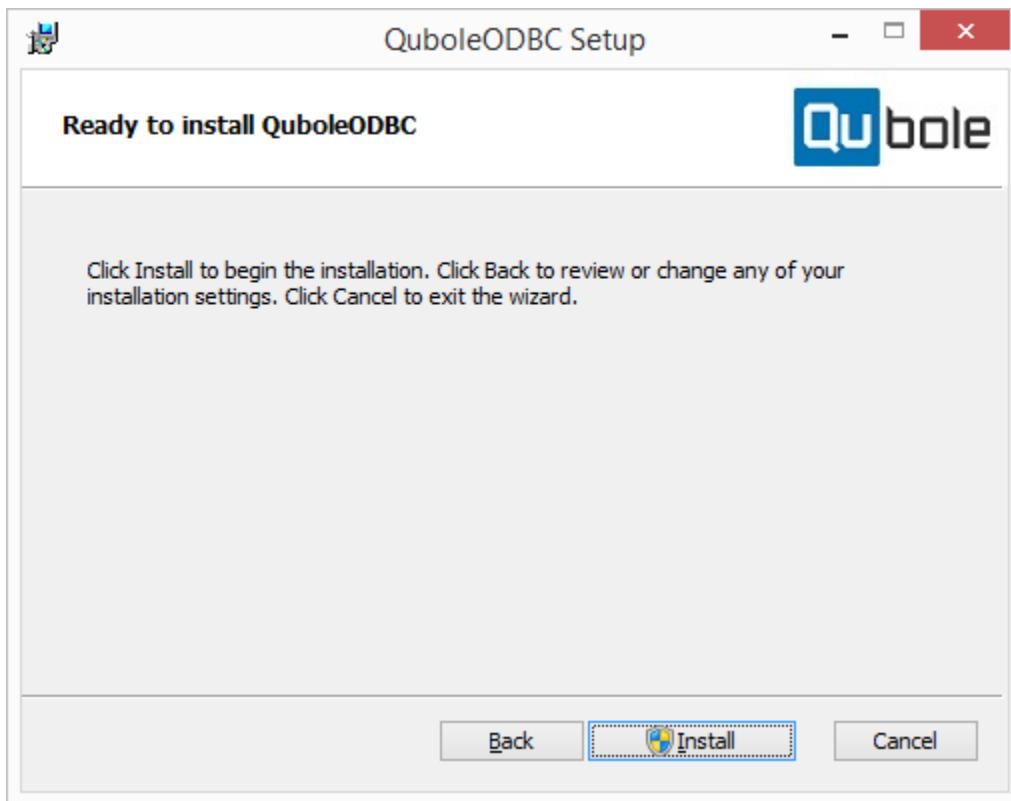
Click **Next** to continue. Click **Back** to go to the previous dialog and click **Cancel** if you want to cancel the installation.

6. Clicking **Next** displays the dialog to enter the cluster labels as shown in the following figure.



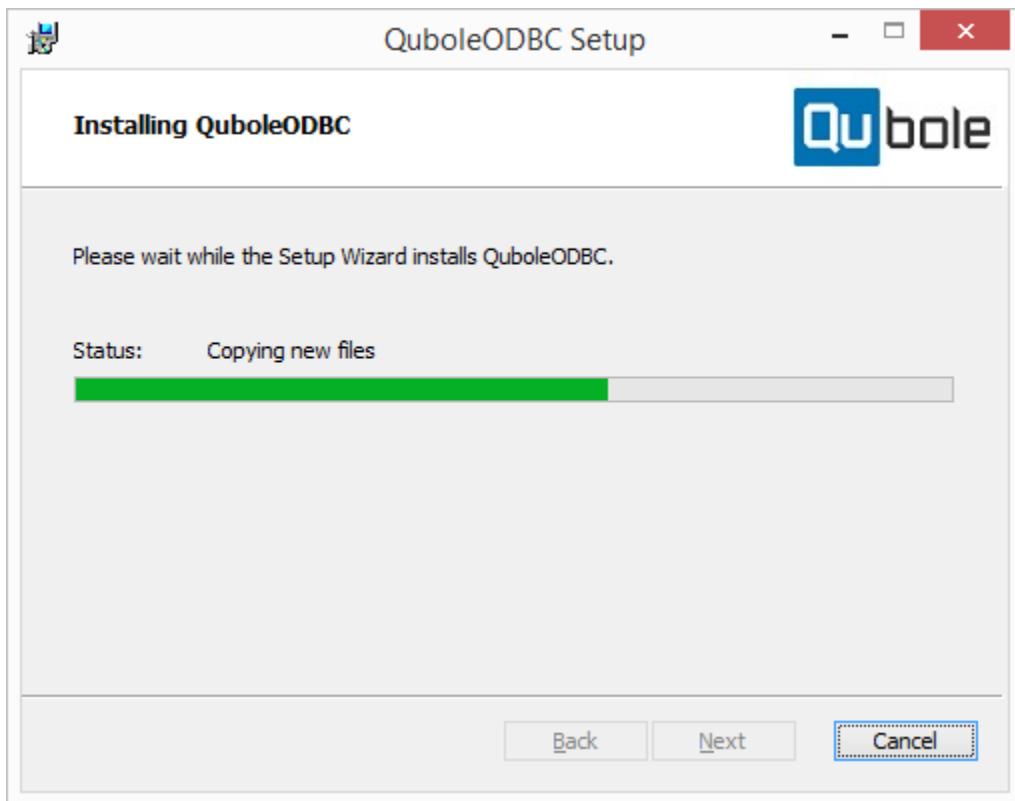
This is an optional setting. You can skip it. Commands run on the default cluster if cluster labels are not specified. Click **Next** to continue. Click **Back** to go to the previous dialog and click **Cancel** if you want to cancel the installation.

7. Clicking **Next** displays the **Ready to Install** dialog as shown in the following figure.

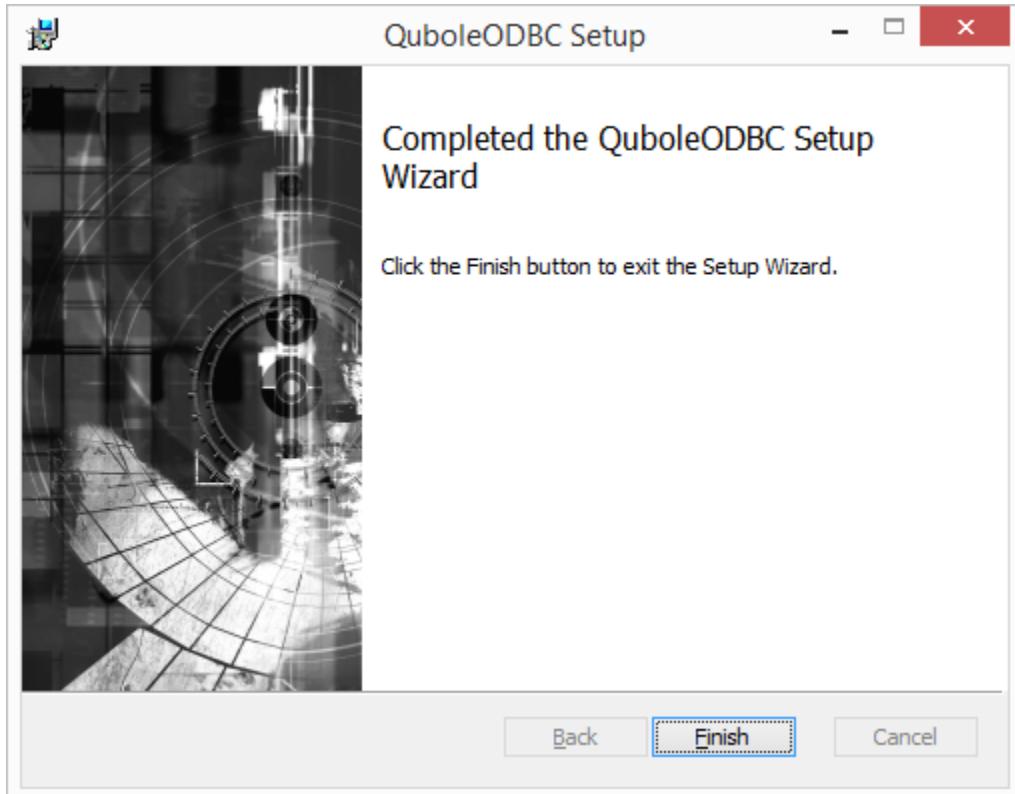


Click **Install** to begin the installation. Click **Back** to go to the previous dialog and click **Cancel** if you want to cancel the installation.

8. The installation process begins and the progress is as shown in the following figure.



After the installation is complete, the message is as shown in the following figure.



Click **Finish** to complete the installation process.

3.9.3 Handling ODBC Driver After Switching User Accounts

If you switch to a different user account, the other account has a different API token. As the API token is account-specific, it is not applicable to the other account (that you have switched to). So, you must change the ODBC administrator in Microsoft Windows' Control Panel. (**Windows > Control Panel > System and Security > Administrative Tools > ODBC Data Sources**).

3.10 QDS Administration How-to Topics

The following topic(s) describe how to perform administration tasks related to QDS:

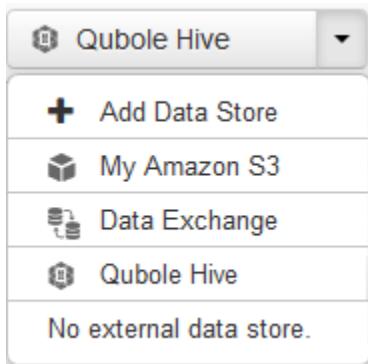
3.10.1 Connect to an AWS Redshift/RDS Instance in a VPC

QDS supports adding data stores to import data and export analyzed data to external databases such as MySQL and Redshift. See [Data Store](#) for more information on data stores. See the [Data Import](#) and [Data Export](#) for more information on importing and exporting data.

To connect to an AWS Redshift/RDS instance in a VPC, create a data store on QDS. See [Adding a Data Store](#) for more information.

Perform the following steps to create a data store at [Explore](#) by performing the following steps:

1. Navigate to [Explore](#) and pull down the drop-down list to see it as shown in the following figure.



Only a system administrator can add a data store and hence, see the Add Data Store option.

By default, it shows **Database Type** as **MySQL** as shown in the following figure.

Connection Details

Database Type	MySQL
Database Name	Database Name
Server Address	Server Address
Port	Port (blank for default)
Username	Username
Password	Password
Region	us-east-1 (North Virginia)

Save **Reset** **Cancel**

* Please ensure that our Amazon account id (805246085872) and our security group ('default') have access privileges to your database.
- If your database is not in VPC then you can give these permissions by adding rules to security group of database.
- If your database is in VPC use 'on-premise' as database region.
See [here](#) for more information.

2. Select the Redshift database type in the **Database Type** if you want a Redshift data store.
3. Enter the name in the **Database Name** text field.
4. Enter the server address in the **Server Address** text field. **Drop an email to help@qubole.com to request for the server address.**
5. Enter the port number in the **Port** text field. You can leave this text field blank for default port numbers.
6. Enter the username of the database in the **Username** text field.
7. Enter the password of the database in the **Password** text field.
8. Select the **on-premise/other** from the AWS **Region** drop-down list as the database is in a VPC. Once you select the on-premise/other region, **Details of gateway machine** options are displayed as shown in the following figure.

Connection Details

Database Type: Redshift

Database Name: Database Name

Server Address: Server Address

Port: Port (blank for default)

Username: Username

Password: Password

Region: on-premise/other

Details of gateway machine

IP address: IP address

Username: Username

Private Key: Private key

Buttons: Save, Reset, Cancel

* On-premise region can be used for: non-AWS databases, databases in AWS vpc or databases in AWS regions not supported by us. To enable us to connect to this database you can use either of following approach:
 - Open your database to entire world.
 - Provide us details of a gateway machine that has access to this database. We will use this gateway machine to connect to your database using secure tunnels.
 If you need more details or help, then please contact us.
 See [here](#) for more information.

9. Enter the IP address of the gateway machine that has access to the RedShift/RDS instance.
10. Type the user name to access the gateway machine in the **Username** text field.
11. Enter the private key to access the gateway machine in the **Private Key** text field.
12. Click **Save** for creating a data store to connect to the Redshift/RDS instance in a VPC. Click **Reset** to reenter the values. Click **Cancel** if you do not want the data store to be created.

3.10.2 Enable Recommissioning on Hadoop-1 and Hadoop-2 Clusters

Qubole supports recommissioning on Hadoop-1 and Hadoop-2 clusters. Recommissioning allows Hadoop-1 and Hadoop-2 clusters to use the current down-scaling nodes if the cluster gets an upscaling request at the time when the nodes are being downscaled.

It is a useful feature as decommissioning and removing a node takes a long time and in the meantime, a new upscaling event can occur.

Recommissioning on clusters is not enabled by default. It can be enabled as an Hadoop override setting:
`mapred.hustler.recommissioning.enabled=true`.
To disable the feature, set `mapred.hustler.recommissioning.enabled=false`.

3.10.3 Run Utility Commands in a Cluster

Utility commands can be run within a cluster to know information about cluster nodes.

- **nodeinfo** is a command to get the cluster node-specific details.

Syntax:

```
nodeinfo [is_master | master_ip | master_public_dns]
```

`nodeinfo is_master` displays if the node is master or slave in boolean format (1/0).

`nodeinfo master_ip` displays the public IP address of the master node.

`nodeinfo master_public_dns` displays the public DNS address of the master node.

Add `$(nodeinfo master_public_dns)` shell script in a node bootstrap file or in any shell script file to get the public DNS hostname of the master node.

See [About Node Bootstrap Script](#) for more information.

To see the public IP address of slave and master nodes, you can also use the QDS UI. Navigate to the Control Panel > Clusters. Click the nodes of the running cluster to see the list of slave nodes and master node with the corresponding public IP address.

Within a specific cluster node, run the `nodeinfo` command to check if the node is a slave or master or get the master node's IP address and public DNS address. A sample illustration of running the `nodeinfo` command is as shown below.

```
[ec2-user@ip-10-111-11-11 /]$ nodeinfo is_master
1
[ec2-user@ip-10-111-11-11 /]$ nodeinfo master_ip
10.111.11.11
[ec2-user@ip-10-111-11-11 /]$ nodeinfo master_public_dns
ec2-54-54-544-544.compute-1.amazonaws.com
[ec2-user@ip-10-111-11-11 /]$
```

3.10.4 Set S3 Optimizations

The following topics describe how to set S3 optimizations.

Enable Multipart Upload and Streaming

Uploading larger objects to an S3 location can be simplified using a multipart upload or multipart streaming option.

Enable Multipart Upload

Multipart upload enables you to upload larger objects in several parts.

`fs.s3n.multipart.uploads.enabled` is the property to enable and disable multipart upload.

The following properties are associated with the multipart upload:

- `fs.s3n.multipart.uploads.enabled` to be set to **true** for enabling multipart upload.
- `fs.s3n.multipart.uploads.enabled` to be set to **false** for disabling multipart upload.
- `fs.s3n.multipart.uploads.maxpartsize.mb` to control the object's part size. Its default value is 500MB.

Enable Multipart Streaming

Multipart streaming uploads files to the Amazon S3 location. Data is uploaded in streams by separating it into chunks. Such chunks are concurrently uploading into the S3 location.

The following properties are associated with the multipart streaming:

- Both `fs.s3n.multipart.uploads.enabled` and `fs.s3n.multipart.uploads.streaming.enabled` must be set to **true** for enabling multipart streaming.
- `fs.s3n.multipart.streaming.uploads.maxpartsize.mb` is the property to control the part size. Its default value is 5 MB.
- `fs.s3n.multipart.uploads.concurrency.factor` is the property to control the number of concurrent parts that can be uploaded. Its default value is 1.

Enable S3 Listing and Wild Card Optimization

Listing files in the S3 location can be a slow process that can be optimized using a configuration option. Similarly, listing directories that contains wildcards can be also slow that can be made faster using a configuration option.

Enable S3 Listing Optimization

As part of the split computation, the Hive must list all files in the table's S3 location. Hadoop jobs must list all files in an S3 location. The implementation in Apache Hadoop for listing files in S3 is very slow. Optimizations have been incorporated to speed this up by setting `fs.s3.inputpathprocessor=true` by default for Hive queries and Hadoop jobs.

Enable WildCard Optimization

There are two forms of wildcard character: asterisk (*) that matches zero or more arbitrary characters and question mark (?) that matches exactly one arbitrary character.

While listing directories that contain an asterisk, for example, `s3://my-bucket/my-dir/*file.csv`, the process may become slow. To speed up the listing process, set `mapred.job.natives3filesystem.globstatus.use` to **true**.

3.10.5 Set Data Compression on Hadoop-1 Clusters

Data compression in Hadoop can speed up the input/output operations as Hadoop jobs are data-intensive. It saves data storage space and makes the data transfer faster over a network. However, there is an increase in CPU utilization and processing time when data is compressed and decompressed. Data Compression and the format used for compressing data have a considerable impact on MapReduce jobs' performance.

Configuring various formats of data compression are as explained below:

- **gzip compression format** - The file extension of this compression format is .gz. This format is not splittable. The following configuration is used to set this format:

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.GzipCodec;
```

- **bzip2 compression format** - The file extension of this compression format is .bz2. This format is splittable. The following configuration is used to set this format:

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.BZip2Codec;
```

- **lzo compression format** - The file extension of this compression format is .lzo. This format is splittable if the compression is indexed. The following configuration is used to set this format:

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=com.hadoop.compression.lzo.LzopCodec;
```

- **snappy compression format** - The file extension of this compression format is .snappy. This format is splittable. The following configuration is used to set this format:

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;
SET mapred.output.compression.type=BLOCK;
```

Example

```
DROP TABLE IF EXISTS manager;
CREATE EXTERNAL TABLE manager( manageid string,yearid string,teamid string) ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',' STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat' LOCATION 's3n://qubole-
DROP TABLE IF EXISTS manager_snappy;
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.SnappyCodec;
SET mapred.output.compression.type=BLOCK;
CREATE TABLE manager_snappy like manager;
INSERT OVERWRITE TABLE manager_snappy
SELECT * FROM manager;
SELECT * FROM manager_snappy limit 3;
```

- **zlib/deflate compression format** - It is the default data compression format. The file extension of this compression format is .deflate. This format is not splittable. The following configuration is used to set this format:

```
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.DefaultCodec;
```

Example

```
DROP TABLE IF EXISTS manager;
CREATE EXTERNAL TABLE manager( manageid string,yearid string,teamid string) ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',' STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat' LOCATION 's3n://qubole-
DROP TABLE IF EXISTS manager_zlib_is_default;
SET hive.exec.compress.output=true;
SET mapred.output.compression.codec=org.apache.hadoop.io.compress.DefaultCodec;
CREATE TABLE manager_zlib_is_default like manager;
INSERT OVERWRITE TABLE manager_zlib_is_default
SELECT * FROM manager;
SELECT * FROM manager_zlib_is_default limit 3;
```

3.10.6 Set up a New Account with QDS

AWS Quick Start Guide describes how to sign up on QDS as a user. Once you sign up as a user, you can access the QDS user interface (UI) to change the account settings and select the authentication type.

QDS supports AWS IAM-based keys and roles authentication for each account on QDS. See *Managing Account Settings* for more information.

Authorizing a QDS Account using AWS IAM Access Keys

See *Authorizing AWS using IAM Keys* to configure a QDS account for getting it authenticated using AWS IAM Access Keys. *Managing Account Settings* describes how to configure and modify a QDS account's settings.

Authorizing a QDS Account using AWS IAM Role Credentials

See *Authorizing AWS using IAM Roles* to configure a QDS account for getting it authenticated using AWS IAM Role credentials. *Managing Account Settings* describes how to configure and modify a QDS account's settings.

Qubole REST API Reference

4.1 Overview

The Qubole Data Service (QDS) is accessible via REST APIs.

To write and test applications that use these APIs, users can use any HTTP client in any programming language to interact with Qubole Data Service. The detailed syntax of the calls is described in subsequent sections.

Users can also use the [QDS Python SDK](#) or the [QDS Java SDK](#) to interact with the Qubole Data Service. The Python SDK also provides an easy-to-use command-line interface.

4.2 Access URL

All URLs referenced in the documentation have the following base:

`https://api.qubole.com/api/${V} /`

where `${V}` refers to the version of the API. Valid values of `${V}` are currently `v1.2` and `latest`.

cURL is a useful tool for testing out Qubole REST API calls from the command line.

The Qubole API is served over both HTTP and HTTPS. When using HTTPS, please ensure that the certificates on the client machine are up-to-date. With an invalid local certificate, you could use the `--insecure` option with cURL to have the call succeed.

4.3 Authentication

API calls must be authenticated with a **Qubole API Token**.

Click on the *Control Panel* in the left pane and navigate to the *My Accounts* tab. Then click on *Show* for the Account for which you want to generate the random API token.

Set the value of this API token to the `AUTH_TOKEN` environment variable when running the API examples via `curl`.

4.4 API Types

The Qubole REST APIs are broadly divided into the following categories:

4.4.1 Command API

The Command APIs let you submit queries and commands, check the status of commands, retrieve results and logs, or cancel commands. The Qubole Data Service currently supports these command types:

- *Hive commands*
- *Hadoop jobs*
- *Pig commands*
- *Presto commands*
- *Spark commands*
- *DbImport commands*
- *DbExport commands*
- *Shell commands*

4.4.2 Hive Metadata API

These APIs provide a set of read-only views that describe your Hive tables and their metadata.

4.4.3 Cluster API

The Cluster APIs let you launch and terminate clusters, and get information about a running cluster. These APIs are meant for advanced users.

4.4.4 DbTap API

A DbTap identifies an external end point for import/export of data from QDS, such as a MySQL instance. The DbTap APIs let you create, view, edit or delete a DbTap.

4.4.5 Scheduler API

The Scheduler APIs let you schedule any command or workflow to run at regular intervals.

4.4.6 Reports API

These APIs let you view aggregated statistical and operational data for your commands.

4.5 Command API

4.5.1 Command Object

Many Command APIs like submitting, canceling or viewing the status, return a (json encoded) command object. The fields of this object are detailed below:

Field	Description
id	The ID of the command.
status	The status of the command can be one of the following: <ul style="list-style-type: none"> <i>waiting</i>: queued in QDS but not started processing yet <i>running</i>: being processed <i>cancelling</i>: the command is being cancelled in response to a user request <i>cancelled</i>: command is complete and was cancelled by the user <i>error</i>: command is complete and failed <i>done</i>: command is complete and was successful
command_type	The type of the command.

4.5.2 Submit a Hive Command

POST /api/v1.2/commands/

This API is used to submit a Hive query.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
query	Specify Hive query to run. Either query or script_location is required
script_location	Specify a S3 path where the hive query to run is stored. Either query or script_location is required. AWS storage credentials stored in the account are used to retrieve the script file.
com-mand_type	Hive command
label	Cluster label to specify the cluster to run this command
retry	Denotes the number of retries for a job. Valid values of <code>retry</code> are 1, 2, and 3.
macros	Expressions to evaluate macros used in the hive command. Refer to Macros in Scheduler for more details.
sample_size	Size of sample in bytes on which to run the query for test mode.
ap-prox_mode_progress	Value of progress for constrained run. Valid value of float between 0 and 1
ap-prox_mode_max_rt	Constrained run max runtime in seconds
ap-prox_mode_min_rt	Constrained run min runtime in seconds
ap-prox_aggregations	Convert count distinct to count approx. Valid values are bool or NULL

Examples

Goal: Show tables

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
    "query": "show tables;", "command_type": "HiveCommand"
}' \
"https://api.qubole.com/api/${V}/commands"
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "command": {
    "approx_mode": false,
    "approx_aggregations": false,
    "query": "show tables",
    "sample": false
  },
  "qbol_session_id": 0000,
  "created_at": "2012-10-11T16:01:09Z",
  "user_id": 00,
  "status": "waiting",
  "command_type": "HiveCommand",
  "id": 3850,
  "progress": 0,
  "meta_data": {
    "results_resource": "commands\\3850\\results",
    "logs_resource": "commands\\3850\\logs"
  }
}
```

Goal: Create an External Table from data on S3

```
export QUERY="create external table miniwikistats (projcode string, pagename string, pageviews int, k  
curl -X POST -H "X-AUTH-TOKEN:$AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{  
    "query": "$QUERY", "command_type": "HiveCommand"  
}' \n"https://api.qubole.com/api/${V}/commands/"
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "command": {
    "approx_mode": false,
    "approx_aggregations": false,
    "query": "create external table miniwikistats (projcode string, pagename string, pageviews int,
    "sample": false
  },
  "qbol_session_id": 0000,
  "created_at": "2012-10-11T16:44:53Z",
  "user_id": 00,
```

```

    "status": "error",
    "command_type": "HiveCommand",
    "id": 3851,
    "progress": 100,
    "meta_data": {
        "results_resource": "commands\3851\results",
        "logs_resource": "commands\3851\logs"
    }
}
}

```

Goal: Count the number of rows in the table

```

export QUERY="select count(*) as num_rows from miniwikistats;

curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
    "query": "$QUERY", "command_type": "HiveCommand"
}'
"https://api.qubole.com/api/${V}/commands/"

```

Response:

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
    "command": {
        "approx_mode": false,
        "approx_aggregations": false,
        "query": "select count(*) as num_rows from miniwikistats;",
        "sample": false
    },
    "qbol_session_id": 0000,
    "created_at": "2012-10-11T16:54:57Z",
    "user_id": 00,
    "status": "waiting",
    "command_type": "HiveCommand",
    "id": 3852,
    "progress": 0,
    "meta_data": {
        "results_resource": "commands\3852\results",
        "logs_resource": "commands\3852\logs"
    }
}

```

Goal: Run a query stored in a S3 file location

Contents of file in S3

```
select count(*) from miniwikistats
```

Payload

```
{
    "script_location": "<S3 Path>", "command_type": "HiveCommand"
}
```

Request

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: applicat
```

Goal: Run a parameterized query stored in a S3 file location

Contents of file in S3

```
select count(*) from miniwikistats where dt = '$formatted_date'
```

Payload

```
{
    "script_location": "<S3 Path>",
    "macros": [{"date": "moment('2011-01-11T00:00:00+00:00')"}, {"formatted_date": "date.clone().format("},
    "command_type": "HiveCommand"
}
```

Request

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: applicat
```

Take a note of the query id (in this case 3852). It will be used in later examples.

```
export QUERYID=3852
```

Goal: Submitting a Hive Query to a Specific Cluster

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: applicat
```

4.5.3 View Command Status

GET /api/v1.2/commands/ (int: command_id)

Use this API to check the status of any command. A user can check any query for the whole account.

Example

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: applicat
```

Sample response:

```
{
    "command": {
        "approx_mode": false,
        "approx_aggregations": false,
        "query": "select count(*) as num_rows from miniwikistats;",
        "sample": false
    },
    "qbolt_session_id": 0000,
    "created_at": "2012-10-11T16:54:57Z",
    "user_id": 00,
    "status": "done",
    "command_type": "HiveCommand",
    "id": 3852,
```

```

"progress": 100,
"meta_data": {
    "results_resource": "commands\3852\results",
    "logs_resource": "commands\3852\logs"
}
}

```

When checking the status of a Hadoop command, the JSON response object contains an additional field ('job_url'), the value of which is URL to JobTracker page specific to this job. Detailed information related to the job such as the number of mappers and reducers, current status, counters, and so on can be retrieved using this URL.

4.5.4 View Command Results

GET /api/v1.2/commands/ (int: command_id) /results

This retrieves results for a completed command (*command_id*).

Parameters

Parameter	Description
inline	Valid values are <code>true</code> (default) or <code>false</code> . This is a request for the results to be returned inline.

Response

The response will contain either the results inline (TAB separated) or will contain an S3 directory location that contains the actual results.

When the output of the command is less than 20MB and contains less than 700 files, the results are packaged and returned inline.

Status Code 422: Command is not done. Results are unavailable.

Example

Goal

To view the results of command, example QUERYID=1234

```

curl -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Content-Type: application/json" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/commands/$QUERYID/results"

```

Response

The following is the response, if the result is inlined.

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{"inline":true, "results":"1\t240\r\n2\t300"}

```

The following is the response, if the result is NOT inlined:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{"inline":false, "result_location": [ "An array of S3 paths. Directories end with '/' in end" ]}
```

4.5.5 View Command Logs

GET /api/v1.2/commands/ (int: command_id) /logs

Retrieves the log(i.e. stderr) of the command (*command_id*).

Response

The response is raw text containing the log of the command.

Example

Goal

To view the logs of command, example QUERYID=1234

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN"
-H "Content-Type: application/json"
-H "Accept: text/plain"
"https://api.qubole.com/api/v1.2/commands/${QUERYID}/logs"
```

Response

```
Total MapReduce jobs = 1
Getting Hadoop cluster information ...
Cluster not found - provisioning cluster machines ...
Waiting for Hadoop to come up ...
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapred.reduce.tasks=<number>
.....
.....
.....
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 1
2012-10-11 16:57:30,346 Stage-1 map = 0%,  reduce = 0%
2012-10-11 16:57:59,725 Stage-1 map = 1%,  reduce = 0%
2012-10-11 16:58:02,784 Stage-1 map = 8%,  reduce = 0%
.....
.....
.....
2012-10-11 16:59:46,147 Stage-1 map = 99%,  reduce = 0%, Cumulative CPU 73.01 sec
2012-10-11 16:59:47,159 Stage-1 map = 99%,  reduce = 0%, Cumulative CPU 73.01 sec
```

```

2012-10-11 16:59:48,172 Stage-1 map = 99%, reduce = 0%, Cumulative CPU 73.01 sec
2012-10-11 16:59:49,183 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 73.01 sec
2012-10-11 16:59:50,195 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 73.01 sec
2012-10-11 16:59:51,207 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 73.01 sec
2012-10-11 16:59:52,218 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 73.01 sec
.....
.....
.....
2012-10-11 17:00:06,655 Stage-1 map = 100%, reduce = 17%, Cumulative CPU 115.15 sec
2012-10-11 17:00:07,668 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 115.15 sec
2012-10-11 17:00:08,684 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 119.52 sec
MapReduce Total cumulative CPU time: 1 minutes 59 seconds 520 msec
Ended Job = job_14.201210111655_0001
1 Rows loaded to s3n://paid-qubole/.....
MapReduce Jobs Launched:
Job 0: Map: 2 Reduce: 1 Accumulative CPU: 119.52 sec HDFS Read: 0 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 59 seconds 520 msec
OK
Time taken: 303.329 seconds

```

4.5.6 View Hadoop Jobs Spawned By a Command

GET /api/v1.2/commands/ (int: command_id) /jobs

Retrieves the details of the hadoop jobs spawned on the cluster by command (*command_id*). This information is only available for commands, which have been completed.

Response

The response is an array of JSON objects with the following details:

Table 4.1: Job Fields

Field	Description
job_stats	Various details of the job, namely its counters, start time, finish time, and so on.
url	The JobTracker URL for the job.
job_id	The job ID.
error_message	A message indicating an error when retrieving the job details. It is not present if there is no error.
http_error_code	The HTTP error code if any. It is present only if there was an HTTP error retrieving job details.

Example

Goal

To view the status and counters of hadoop jobs spawned by command, *1234*.

```

curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN"
-H "Content-Type: application/json"
-H "Accept: application/json"
"https://api.qubole.com/api/v1.2/commands/1234/jobs"

```

Response

```
[  
{  
    "job_stats": {  
        "finished_at": "Mon Feb 16 07:54:56 UTC 2015",  
        "user": "foo@bar.com",  
        "pool_name": "foo@bar.com",  
        "status": "SUCCEEDED",  
        "job_name": "110163-ShellCommand",  
        "reduce": {  
            "num_tasks": "0",  
            "pending_tasks": "0",  
            "complete_tasks": "0",  
            "complete_percent": "100.0",  
            "killed_task_attempts": "0",  
            "running_tasks": "0",  
            "failed_task_attempts": "0",  
            "killed_tasks": "0"  
        },  
        "counters": {  
            "org.apache.hadoop.mapred.JobInProgress$Counter": [  
                {  
                    "name": "Total time spent by all maps",  
                    "total_value": "4,545",  
                    "reduce_value": "0",  
                    "map_value": "0"  
                },  
                {  
                    "name": "Total time spent by all reduces waiting after reserving slots (ms)",  
                    "total_value": "0",  
                    "reduce_value": "0",  
                    "map_value": "0"  
                },  
                {  
                    "name": "Total time spent by all maps waiting after reserving slots (ms)",  
                    "total_value": "0",  
                    "reduce_value": "0",  
                    "map_value": "0"  
                },  
                {  
                    "name": "Launched map tasks",  
                    "total_value": "1",  
                    "reduce_value": "0",  
                    "map_value": "0"  
                },  
                {  
                    "name": "Total time spent by all reduces",  
                    "total_value": "0",  
                    "reduce_value": "0",  
                    "map_value": "0"  
                }  
            ],  
            "com.qubole.ShellLauncher$MapCounter": [  
                {  
                    "name": "HasActualJobStartedYet",  
                    "total_value": "1",  
                    "reduce_value": "0",  
                    "map_value": "1"  
                },  
            ]  
        }  
    }  
}
```

```

        {
            "name": "ShellExitCode",
            "total_value": "0",
            "reduce_value": "0",
            "map_value": "0"
        }
    ],
    "org.apache.hadoop.mapred.Task$Counter": [
        {
            "name": "Map input records",
            "total_value": "1",
            "reduce_value": "0",
            "map_value": "1"
        },
        {
            "name": "Total physical memory in bytes",
            "total_value": "89,743,360",
            "reduce_value": "0",
            "map_value": "89,743,360"
        },
        {
            "name": "Spilled Records",
            "total_value": "0",
            "reduce_value": "0",
            "map_value": "0"
        },
        {
            "name": "MAP_TASK_WALLCLOCK",
            "total_value": "2,402",
            "reduce_value": "0",
            "map_value": "2,402"
        },
        {
            "name": "Total cumulative CPU milliseconds",
            "total_value": "530",
            "reduce_value": "0",
            "map_value": "530"
        },
        {
            "name": "Map input bytes",
            "total_value": "1",
            "reduce_value": "0",
            "map_value": "1"
        },
        {
            "name": "Total virtual memory in bytes",
            "total_value": "1,122,668,544",
            "reduce_value": "0",
            "map_value": "1,122,668,544"
        },
        {
            "name": "Map output records",
            "total_value": "0",
            "reduce_value": "0",
            "map_value": "0"
        }
    ]
},

```

```
"started_at": "Mon Feb 16 07:54:50 UTC 2015",
"map": {
    "num_tasks": "1",
    "pending_tasks": "0",
    "complete_tasks": "1",
    "complete_percent": "100.0",
    "killed_task_attempts": "0",
    "running_tasks": "0",
    "failed_task_attempts": "0",
    "killed_tasks": "0"
}
},
"url": "https://api.qubole.net/qpal/handle_proxy?query=http%3A%2F%2Fec2-54-161-105-44.compute-1.",
"job_id": "job_11.201502160742_0001"
},
{
"job_stats": {
    "finished_at": "Mon Feb 16 08:02:47 UTC 2015",
    "user": "foo@bar.com",
    "pool_name": "foo@bar.com",
    "status": "SUCCEEDED",
    "job_name": "TeraGen",
    "reduce": {
        "num_tasks": "0",
        "pending_tasks": "0",
        "complete_tasks": "0",
        "complete_percent": "100.0",
        "killed_task_attempts": "0",
        "running_tasks": "0",
        "failed_task_attempts": "0",
        "killed_tasks": "0"
    },
    "counters": {
        "org.apache.hadoop.mapred.JobInProgress$Counter": [
            {
                "name": "Total time spent by all maps",
                "total_value": "6,293,376",
                "reduce_value": "0",
                "map_value": "0"
            },
            {
                "name": "Total time spent by all reduces waiting after reserving slots (ms)",
                "total_value": "0",
                "reduce_value": "0",
                "map_value": "0"
            },
            {
                "name": "Total time spent by all maps waiting after reserving slots (ms)",
                "total_value": "0",
                "reduce_value": "0",
                "map_value": "0"
            },
            {
                "name": "Launched map tasks",
                "total_value": "46",
                "reduce_value": "0",
                "map_value": "0"
            }
        ]
    }
}
```

```

        {
            "name": "Total time spent by all reduces",
            "total_value": "0",
            "reduce_value": "0",
            "map_value": "0"
        }
    ],
    "FileSystemCounters": [
        {
            "name": "HDFS_FILES_CREATED",
            "total_value": "40",
            "reduce_value": "0",
            "map_value": "40"
        },
        {
            "name": "HDFS_BYTES_WRITTEN",
            "total_value": "100,000,000,000",
            "reduce_value": "0",
            "map_value": "100,000,000,000"
        }
    ],
    "org.apache.hadoop.mapred.Task$Counter": [
        {
            "name": "Map input records",
            "total_value": "1,000,000,000",
            "reduce_value": "0",
            "map_value": "1,000,000,000"
        },
        {
            "name": "Total physical memory in bytes",
            "total_value": "5,167,636,480",
            "reduce_value": "0",
            "map_value": "5,167,636,480"
        },
        {
            "name": "Spilled Records",
            "total_value": "0",
            "reduce_value": "0",
            "map_value": "0"
        },
        {
            "name": "MAP_TASK_WALLCLOCK",
            "total_value": "6,066,445",
            "reduce_value": "0",
            "map_value": "6,066,445"
        },
        {
            "name": "Total cumulative CPU milliseconds",
            "total_value": "3,237,210",
            "reduce_value": "0",
            "map_value": "3,237,210"
        },
        {
            "name": "Map input bytes",
            "total_value": "1,000,000,000",
            "reduce_value": "0",
            "map_value": "1,000,000,000"
        }
    ]
}

```

4.5.7 Cancel a Command

PUT /api/v1.2/commands/ (int: *id*)

This API is used to cancel a command.

Parameters

Parameter	Description
status	kill

Example

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

4.5.8 View Command History

This API is to view the command history.

View Queries by User Id

Resource URI	commands/
Request Type	GET
Supporting Versions	v1, v1.2
Return Value	This curl request will return a JSON object containing the Command Objects with all its attributes as described above. Additionally returned JSON object contained the next_page, previous page, and per_page parameter.

Parameters

Parameter	Description
page	Integer
Tuples per page	Integer

Examples:

To get last 10 commands for current user:

```
curl -i -X GET -H "X-AUTH-TOKEN:$AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept:application/json"
```

(Pagination) To get results 10-12 for current user (4th page with 3 results per page):

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

or

```
curl -i -H "X-AUTH-TOKEN: $AUTH_TOKEN" "https://api.qubole.com/api/${V}/commands?page=4&per_page=3"
```

The sample response for the paginated call will be:

```
{
  "paging_info": {"previous_page": 3, "next_page": 5, "per_page": 3},
  "commands": [ {<standard command object as described in create a command>} , ... ]}
```

4.5.9 Submit a Pig Command

POST /api/v1.2/commands/

This API is for submitting a pig command.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
script_location	S3 location of the Pig script. The request must contain either <code>latin_statements</code> or <code>script_location</code>
parameters	JSON hash of Pig params.
latin_statements	PigLatin statements to execute. The request must contain either <code>latin_statements</code> or <code>script_location</code>
com-mand_type	PigCommand
label	Cluster label to specify the cluster to run this command
retry	Denotes the number of retries for a job. Valid values of <code>retry</code> are 1, 2, and 3.

Response

A JSON object representing the newly created command.

Examples

Files Used In The Examples

These files are cloned from [The Apache Pig Tutorial](#)

Example Name	Location
Dataset	s3://paid-qubole/PigAPIDemo/data/excite-small.log
Parametrized Pig Script	s3://paid-qubole/PigAPIDemo/scripts/script1-hadoop-parametrized.pig
Java UDF jar	s3://paid-qubole/PigAPIDemo/jars/tutorial.jar
Pig Script	s3://paid-qubole/PigAPIDemo/scripts/script1-hadoop-s3-small.pig

Sample Request – Non-Parametrized script

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" \
-H "Accept: application/json" \
-d '{
"script_location": "s3://paid-qubole/PigAPIDemo/scripts/script1-hadoop-s3-small.pig",
"command_type": "PigCommand"}'  "https://api.qubole.com/api/v1.2/commands"
```

Sample Request – Parametrized script

```
export $output_location=<your s3 output location>

curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" \
-H "Accept: application/json" \
-d '{
"command_type": "PigCommand",
"parameters": {
  "output": "s3://paid-qubole/PigAPIDemo/output",
  "input": "s3://paid-qubole/PigAPIDemo/data/excite-small.log",
  "udf_jar": "s3://paid-qubole/PigAPIDemo/jars/tutorial.jar"
},
"script_location": "s3://paid-qubole/PigAPIDemo/scripts/script1-hadoop-parametrized.pig"
}' "https://api.qubole.com/api/v1.2/commands"
```

Sample Response – Non-Parametrized script

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "timeout": null,
  "id": 283032,
  "path": "/tmp/2014-07-14/235/283032",
  "end_time": null,
  "resolved_macros": null,
  "start_time": null,
  "name": null,
  "label": null,
  "meta_data": {
    "results_resource": "commands/283032/results",
    "logs_resource": "commands/283032/logs"
  },
  "can_notify": false,
  "nominal_time": null,
  "command": {
    "latin_statements": null,
    "script_location": "s3://paid-qubole/PigAPIDemo/scripts/script1-hadoop-s3-small.pig",
    "parameters": null
  },
  "command_type": "PigCommand",
  "pool": null,
  "user_id": 846,
  "num_result_dir": -1,
  "status": "waiting",
  "pid": null,
  "qlog": null,
  "created_at": "2014-07-14T06:53:08Z",
  "sequence_id": null,
  "submit_time": 1405320788,
  "progress": 0,
  "template": "generic",
  "qbol_session_id": 38395
}

```

Sample Request – Using latin_statements

For small scripts, it's usually convenient to inline the script:

```

curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" \
-H "Accept: application/json" \
-d '{"latin_statements":"A = LOAD \\"s3://paid-qubole/PigAPIDemo/data/excite-small.log\\";
"command_type":"PigCommand"}' "https://api.qubole.com/api/v1.2/commands" | dump A; \

```

Even with short scripts, latin_statements in curl request can get difficult to construct due to escaping issues. Instead, create a local file – say “script.pig” and copy the script there and try this.

Sample Request – Providing a local script file in the request.

```

curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" \
--data-urlencode latin_statements@script.pig -d command_type=PigCommand \
"https://api.qubole.com/api/v1.2/commands"

```

Note that all input/output data and the UDF jars must be present in S3 bucket.

4.5.10 Submit a Hadoop Jar Command

POST /api/v1.2/commands/

This API is used to submit a Hadoop Jar command.

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
command_type	Hadoop command
sub_command	jar
sub_command_args	s3_path_to_jar [main_class] [hadoop-generic-option]
label	Cluster label to specify the cluster to run this command
retry	Denotes the number of retries for a job. Valid values of retry are 1, 2, and 3.

Examples

The example given below runs a Hadoop Streaming job. The streaming jar is stored on S3 and the application just runs a map-only job running the Unix utility *wc* against the input dataset.

Hadoop Streaming Job

```
export OUTPUT_LOC=<s3 output location>;
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Sample Response

```
{
  "id":4246,
  "meta_data":
  {
    "results_resource":"commands/4246/results",
    "logs_resource":"commands/4246/logs"
  },
  "command":{ "sub_command_args":"s3n://paid-qubole/HadoopAPITests/jars/hadoop-0.20.1-dev-streaming.jar wc /tmp/input /tmp/output" },
  "progress":0,
  "status":"waiting",
  "command_type":"HadoopCommand",
  "qbol_session_id":1629,
  "created_at":"2012-10-16T11:29:36Z",
  "user_id":9
}
```

Hadoop Jar Gutenberg Job

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Hadoop Streaming Job with a Cluster Label

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Hadoop Streaming Job without a Cluster Label

Note: When a job is run without a cluster label, the default cluster runs the command.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

4.5.11 Submit a Spark Command

POST /api/v1.2/commands/

This API is used to submit a Spark command.

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameters

Parameter	Description
program	Provide the complete Spark Program in Scala, SQL, Command, R, or Python.
language	Specify the language of the program, Scala, SQL, Command, R, or Python. Required only when a program is used.
arguments	Specify the spark-submit command line arguments here.
user_program_arguments	The arguments that the user program takes in.
cmdline	Alternatively, you can provide the spark-submit command line itself. If you use this option, you cannot use any other parameters mentioned here. All required information is captured in command line itself.
command_type	Spark command
label	Cluster label to specify the cluster to run this command
app_id	ID of an app, which is a main abstraction of the Spark Job Server API. An app is used to store the configuration for a Spark application. See Understanding the Spark Job Server for more information.

Examples

Examples are written in python and uses pyCurl. Using CURL directly is possible but hard as the program needs escaping. Also, JSON does not support new lines. To avoid confusion, these python API examples are provided which are clear and can be used directly.

Alternatively, you can use qds-sdk-py directly.

Example Python API Framework

```
import sys
import pycurl
import json
c= pycurl.Curl()
url="https://api.qubole.com/api/v1.2/commands"
auth_token = <provide auth token here>
c.setopt(pycurl.URL, url)
c.setopt(pycurl.HEADER, ["X-AUTH-TOKEN: "+ auth_token, "Content-Type:application/json", "Accept:application/json"])
c.setopt(pycurl.POST, 1)
```

(After this, select any of the following examples depending on the requirement.)

The above code snippet can be used to make API calls. The following examples uses the above program as its base and shows various use-cases.

Example to Submit Spark Scala Program

```
prog = '''
import scala.math.random

import org.apache.spark._

/** Computes an approximation to pi */
object SparkPi {
  def main(args: Array[String]) {
    val conf = new SparkConf().setAppName("Spark Pi")
    val spark = new SparkContext(conf)
    val slices = if (args.length > 0) args(0).toInt else 2
    val n = math.min(100000L * slices, Int.MaxValue.toInt // avoid overflow
    val count = spark.parallelize(1 until n, slices).map { i =>
      val x = random * 2 - 1
      val y = random * 2 - 1
      if (x*x + y*y < 1) 1 else 0
    }.reduce(_ + _)
    println("Pi is roughly " + 4.0 * count / n)
    spark.stop()
  }
}
'''

data=json.dumps({"program":prog,"language":"scala","arguments":"--class SparkPi", command_type:"SparkJobServer", "label":"spark", "app_id": "3"})

c.setopt(pycurl.POSTFIELDS, data)
c.perform()
```

To submit a snippet to the Spark Job Server app, use the following data payload instead of the above data.

```
data=json.dumps({"program":prog,"language":"scala","arguments":"--class SparkPi", command_type:"SparkJobServer", "label":"spark", "app_id": "3" })
```

Where app_id = Spark Job Server app ID. See [Understanding the Spark Job Server](#) for more information.

Example to Submit Spark Python Program

Here is the Spark Pi example in Python.

```

prog = """
import sys
from random import random
from operator import add

from pyspark import SparkContext

if __name__ == "__main__":
    """
        Usage: pi [partitions]
    """
    sc = SparkContext(appName="PythonPi")
    partitions = int(sys.argv[1]) if len(sys.argv) > 1 else 2
    n = 100000 * partitions

    def f(_):
        x = random() * 2 - 1
        y = random() * 2 - 1
        return 1 if x ** 2 + y ** 2 < 1 else 0

    count = sc.parallelize(xrange(1, n + 1), partitions).map(f).reduce(add)
    print "Pi is roughly %f" % (4.0 * count / n)

    sc.stop()
"""

data=json.dumps({"program":prog,"language":"python","command_type":"SparkCommand"})

c.setopt(pycurl.POSTFIELDS, data)
c.perform()

```

Example to Add Spark Submit Options

Add arguments in JSON body to supply spark-submit options.

```

data=json.dumps(
{"program":prog,
"language":"python", "arguments": "--num-executors 10 --max-executors 10 --executor-memory 5G --execu
"command_type":"SparkCommand"})

```

Example to Add Arguments to User Program

Add user_program_arguments in JSON body. Here is a sample program which takes in arguments (input and output location).

```

prog='''
import org.apache.spark._
object testprogram {
  def main(args: Array[String]) {
    var sc = new SparkContext(new SparkConf())
    var fileToRead = args(0)//passed as args in user program
    var fileToWrite = args(1)
    var output = sc.textFile(fileToRead).saveAsTextFile(fileToWrite)
    println(output)
  }
}

```

```
}'''  
data=json.dumps(  
{ "program":prog,  
"language":"scala",  
"arguments" "--class testprogram",  
"user_program_arguments": "s3://bucket/path/to/source s3://bucket/path/to/destination",  
"command_type":"SparkCommand", })  
  
c.setopt(pycurl.POSTFIELDS, data)  
c.perform()
```

Example to Use Command Line Parameter

For power users, Qubole provides the ability to provide the spark-submit command line directly. This is explained in detail [here](#).

In this case, you must compile the program (in case of Scala), create a jar, upload the file to S3 and invoke the command line. Note that Qubole's deployment of Spark is available at the /usr/lib/spark directory: /usr/lib/spark/bin/spark-submit [options] <app jar in s3 | python file> [app options].

Example to Submit Spark Command in SQL

You can submit a Spark Command in SQL. Here is an example to submit a Spark Command in SQL.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{  
    "sql": "select * from default_qubole_memetracker limit 10;",  
    "language": "sql", "command_type": "SparkCommand", "label": "spark"  
}'  
"https://api.qubole.com/api/${V}/commands"
```

Example to Submit a Spark Command in SQL to a Spark Job Server App

You can submit a Spark command in SQL to an existing Spark Job Server app.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{  
    "sql": "select * from default_qubole_memetracker limit 10;",  
    "language": "sql", "command_type": "SparkCommand", "label": "spark", "app_id": "3"  
}'  
"https://api.qubole.com/api/${V}/commands"
```

Where app_id = Spark Job Server app ID. See [Understanding the Spark Job Server](#) for more information.

4.5.12 Submit a Hadoop S3DistCp Command

POST /api/v1.2/commands/

Hadoop DistCP is the tool used for copying large amount of data across clusters. S3DistCp is an extension of DistCp that is optimized to work with Amazon Web Services (AWS).

In Qubole context, if you are running multiple jobs on the same datasets, then S3DistCp can be used to copy large amounts of data from S3 to HDFS. Subsequent jobs can now point to the data in HDFS location directly. You can also use S3DistCp to copy the data from HDFS to S3. For more details on S3DistCp, see [here](#).

Parameter	Description
command_type	HadoopCommand
sub_command	s3distcp
sub_command_args	[hadoop-generic-options] [s3distcp-args] [s3distcp-args]

Example

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Remember to change the source folder before running the above command. Source folder must be some S3 location, which is accessible using your AWS credentials.

Sample Response

```
{
  "id": 18167,
  "meta_data": {
    "logs_resource": "commands/18167/logs",
    "results_resource": "commands/18167/results"
  },
  "command": {
    "sub_command": "s3distcp",
    "sub_command_args": "--src s3://paid-qubole/kaggle_data/HeritageHealthPrize/ --dest /datasets"
  },
  "command_type": "HadoopCommand",
  "created_at": "2013-03-14T09:34:15Z",
  "path": "/tmp/2013-03-14/53/18167",
  "progress": 0,
  "qbol_session_id": 3525,
  "qlog": null,
  "resolved_macros": null,
  "status": "waiting"
}
```

4.5.13 Submit a DB Import Command

POST /api/v1.2/commands/

Import commands allows users to pull data from a relational database to QDS in a hive table. User can either pull a complete table or a subset of it and this can be done using mode 1. In mode 1, to selectively pull the rows, the user can specify the condition using db_where option. If only a subset of column must be pulled, then the user can specify a comma separated list in db_columns option.

To speed up the process of pulling data, user can set db_parallelism in which case, Qubole opens multiple connections to the database to pull the data. In case of mode 1, Qubole automatically detects and uses primary key column of the table to split the work. Bear in mind that setting a very high value for db_parallelism will pressure the database.

Sophisticated users can specify a custom query to transform the data before pulling it using mode 2. In case of mode 2, if db_parallelism > 1, Qubole needs more input to split the work. So, the user is also required to specify:

- db_split_column: the column which will be used to split the work
- db_boundary_query: which will be used to get a range on the column typically it will be something like:
select min(db_split_column), max(db_split_column) from db_table
- db_extract_query: must have a where clause with \$CONDITIONS in it. At execution time, \$CONDITIONS is replaced by clause like: db_split_column > lower_val AND db_split_column <= upper_val This is used to determine boundary for splitting work.

Simple Mode

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
com-mand_type	DbImport command
hive_table	Specify Hive table to import into
hive_serde	Specify the serde type of the hive table to import into. Supported values: avro and orc. By default, hive table is assumed to be of TXT format.
mode	1
dbtap_id	DbTap ID of the source database in Qubole
db_table	Table to import from
retry	Denotes the number of retries for a job. Valid values of retry are 1, 2, and 3.
db_where	<i>where</i> clause to be applied to the table before extracting rows to be imported
db_parallelism	Number of parallel threads to use for extracting data

Example

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Advanced Mode

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
com-mand_type	DbImport command
hive_table	Specify Hive table to import into
hive_serde	Specify the serde type of the hive table to import into. Supported values: avro and orc. By default, hive table is assumed to be of TXT format.
mode	2
dbtap_id	DbTap ID of the source database in Qubole
db_extract_query	SQL query to be applied at the source database for extracting data. \$CONDITIONS must be part of the where clause.
db_boundary_query	Query to be used get range of row IDs to be extracted
db_split_column	Column used as row ID to split data into ranges
db_parallelism	Number of parallel threads to use for extracting data

Example

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

4.5.14 Submit a DB Export Command

POST /api/v1.2/commands/

Export commands allows users to push data from QDS to a relational database. User can either push a particular partition of a table (for partitioned table) or a complete unpartitioned table using mode 1. The `hive_table` and `partition_spec` options can be used to push data.

Export command can also be used to export an HDFS directory or a S3 location using mode 2. Option `export_dir` can be used for this. In this case, user also needs to specify separator used in the data. This can be specified using option `fields_terminated_by`.

On the database side, by default, the data being exported will be appended to the table. However, if user wants to update existing data or update existing data with insert for non-existing rows, then the user should specify this using `db_update_mode` option and give us a comma separated list of columns using `db_update_keys`. This is used to determine uniqueness of a row. It uses upsert functionality provided by the underlying database and is supported only for [MySQL](#) and [Oracle](#).

Exporting data from Hive

Parameter	Description
com-mand_type	The DbExport command.
mode****	1
hive_table	The name of the Hive Table from which data will be exported.
db-tap_id****	The data store ID of the target database, in Qubole.
db_table	The target database table to export to.
retry	Denotes the number of retries for a job. Valid values of <code>retry</code> are 1, 2, and 3.
parti-tion_spec	The partition specification for Hive table.
db_update	This can be <i>allowinsert</i> or <i>updateonly</i> . If <i>updateonly</i> is specified, only the existing rows are updated. If <i>allowinsert</i> is specified, then existing rows are updated and non existing rows are inserted. If this option is not specified, then the given the data will be appended to the table.
db_update_ke	The columns used to determine the uniqueness of rows

Example: Exporting an Unpartitioned Hive Table

```
curl -i -X POST -H "X-AUTH-TOKEN: SAUTH TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Assuming a table exported airline origin destination with following schema already exists:

```
CREATE TABLE `exported airline origin destination`(`ItinID` varchar(1000) DEFAULT NULL, `MktID` var
```

Example: Exporting a single Hive Partition

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Assuming a table exported_minitest with following schema already exists:

```
CREATE TABLE `exported_minitest` (`projcode` varchar(100) DEFAULT NULL, `pagename` varchar(1000) DEF
```

Exporting data from HDFS/S3

Parameter	Description
com-command_type	The DbExport command.
mode****	2
export_dir	The HDFS/S3 location from which data will be exported.
db-tap_id****	The data store ID of the target database, in Qubole.
db_table	The target database table to export to.
db_update_mode	This can be <i>allowinsert</i> or <i>updateonly</i> . If <i>updateonly</i> is specified, only the existing rows are updated. If <i>allowinsert</i> is specified, then existing rows are updated and non existing rows are inserted. If this option is not specified, then the given data will be appended to the table.
db_update_keys	The columns used to determine the uniqueness of rows.
fields_terminated_by	Hex of the char used as column separator in the dataset, for example: \0x20 for space.

Example: Exporting from a HDFS/S3 location

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Assuming a table exported_minitest2 with following schema already exists:

```
CREATE TABLE `exported_minitest2` ( `projcode` varchar(100) DEFAULT NULL, `pagename` varchar(1000) DEF
```

Known Issue: Does not work if the database column name has special chars like space and ‘.

4.5.15 Submit a Presto Command

POST /api/v1.2/commands/

This API is used to submit a Presto command.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
query	Specify Presto query to run. Either the query or the script_location is required.
script_location	Specify a S3 path where the presto query to run is stored. Either the query or the script_location is required. The AWS storage credentials stored in the account are used to retrieve the script file.
command_type	Presto command
label	Cluster label to specify the cluster to run this command
retry	Denotes the number of retries for a job. Valid values of retry are 1, 2, and 3.

Examples

Show tables

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Sample Response

```
{
  "command": {
    "query": "show tables",
    "qbol_session_id": 0000,
    "created_at": "2014-01-21T16:01:09Z",
    "user_id": 00,
    "status": "waiting",
    "command_type": "PrestoCommand",
    "id": 4850,
    "progress": 0,
    "meta_data": {
      "results_resource": "commands/4850/results",
      "logs_resource": "commands/4850/logs"
    }
  }
}
```

Count Number of Rows

```
export QUERY="select count(*) as num_rows from miniwikistats;" curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Sample Response

```
{
  "command": {
    "query": "select count(*) as num_rows from miniwikistats;",
    "qbol_session_id": 0000,
    "created_at": "2014-10-11T16:54:57Z",
    "user_id": 00,
    "status": "waiting",
    "command_type": "PrestoCommand",
    "id": 4852,
    "progress": 0,
    "meta_data": {
      "results_resource": "commands/4852/results",
      "logs_resource": "commands/4852/logs"
    }
  }
}
```

4.5.16 Submit a Shell Command

POST /api/v1.2/commands/

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
inline	Inline script to run or submit a Shell command. Either script or script_location is required.
script_location	Specify a S3 path where the shell query to run is stored. Either query or script_location is required. AWS storage credentials stored in the account are used.
files	List of files in s3 bucket. Format : file1,file2. These files will be copied to the working directory where the command is executed.
archive	List of archives in s3 bucket. Format : archive1,archive2. These are unarchived in the working directory where the command is executed.
com-mand_type	Shell command
macros	Expressions to evaluate macros used in the shell command. Refer to <i>Macros in Scheduler</i> for more details.
label	The label of the cluster to run the command.
can_notify	Sends an email on command completion.
tags	comma-separated list of tags to be associated with the query

Examples

Goal: Inline script

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
-d '{
    "inline": "hadoop dfs -lsr s3://paid-qubole/;",
    "command_type": "ShellCommand"
}' \
"https://api.qubole.com/api/${V}/commands"
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
    "qlog": null,
    "created_at": "2015-01-12T11:50:21Z",
    "status": "waiting",
    "meta_data": {
        "results_resource": "commands/36/results",
        "logs_resource": "commands/36/logs"
    },
    "account_id": "1",
    "user_id": 1,
    "pool": null,
    "submit_time": 1421063421,
```

```

"progress":0,
"template":"generic",
"pid":null,
"resolved_macros":null,
"label":"default",
"timeout":null,
"can_notify":false,
"qbol_session_id":7,
"command_source":"API",
"name":null,
"num_result_dir":-1,
"end_time":null,
"start_time":null,
"path":"/tmp/2015-01-12/1/36",
"id":36,
"command_type":"ShellCommand",
"command":{
  "files":null,
  "parameters":null,
  "script_location":null,
  "inline":"hadoop dfs -lsr s3://paid-qubole/;",
  "archives":null
}
}
}

```

Goal: Script_location

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
  "script_location":"s3://paid-qubole/ShellDemo/data/excite-small.sh", "command_type":"ShellCommand"
}' \
"https://api.qubole.com/api/${V}/commands"
```

Goal: Running shell commands using Files

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
  "inline":"hadoop dfs -lsr s3://paid-qubole/;", "files":"s3://paid-qubole/ShellDemo/data/excite-small.sh"
}' \
"https://api.qubole.com/api/${V}/commands"
```

Goal: Running shell commands using Archives

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
  "inline":"hadoop dfs -lsr s3://paid-qubole/;", "archives":"s3://paid-qubole/ShellDemo/data/excite-small.sh"
}' \
"https://api.qubole.com/api/${V}/commands"
```

Goal: Using Macros in a shell command

```
curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{
  "inline" : "hadoop dfs -lsr s3://${location}/;", "command_type" : "ShellCommand",
```

```
"macros" : [{"location" : "\"paid-qubole\""}]}' \\\n"https://api.qubole.com/api/${V}/commands"
```

Take a note of how the double quotes are used in the above query.

4.5.17 Submit a DB Tap Query Command

POST /api/v1.2/commands/

This API is to submit a DB Tap query.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
query	Specify DB Tap query to run.
db_tap_id	Specify the DB Tap id of the target database to run the query on.
com-mand_type	DbTapQueryCommand
macros	Expressions to evaluate macros used in the DB Tap Query command. Refer to <i>Macros in Scheduler</i> for more details.

Example

Goal: Show tables

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "timeout": null,
  "qlog": null,
  "status": "waiting",
  "meta_data": {
    "results_resource": "commands/000000/results",
    "logs_resource": "commands/000000/logs"
  },
  "account_id": 00,
  "user_id": 00,
  "command_source": "API",
  "command": {
    "db_tap_id": 1,
    "query": "show tables",
    "md_cmd": null
  }
},
```

```

"pool":null,
"can_notify":false,
"end_time":null,
"command_type":"DbTapQueryCommand",
"label":"default",
"pid":null,
"progress":0,
"num_result_dir":-1,
"created_at":"2014-12-24T09:01:27Z",
"submit_time":1419411687,
"name":null,
"start_time":null,
"template":"generic",
"resolved_macros":null,
"path":"/tmp/2014-12-24/234/000000",
"id":000000,
"qbol_session_id":null
}

```

4.5.18 Submit a Workflow Command

POST /api/v1.2/commands/

This API is used to submit a workflow command. A workflow command contains an array of sub-commands to run as part of a single command.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
sub_commands	say of sub-commands to run as part of this command. Commands will be executed in the sequence they are specified in. Options specific to command types (for example, <i>query</i> for hive commands or <i>inline</i> for shell commands) can be provided in the individual sub-command definitions.
command_type	Composite command

Examples

Goal: Run the terasort benchmark

```

curl -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -d '{
  "sub_commands": [
    {
      "inline": "if hadoop fs -ls /user/hduser/terasort-input; then\n      hadoop fs -rmr /user/hduser/terasort-output;\n    fi",
      "command_type": "ShellCommand"
    },
    {
      "inline": "#!/bin/bash\n\nNUM_MAP_TASKS=20\n\n# Total data generated = (DATA_SIZE * 100) bytes\n",
      "command_type": "ShellCommand"
    }
  ]
}'

```

```
{  
    "inline": "#!/bin/bash\n\nNUM_REDUCE_TASKS=20\nNUM_MAP_TASKS=20\nhadoop jar /usr/lib/hadoop/hadoop-mapreduce-examples.jar terasort 1000 1000 1000  
    "command_type": "ShellCommand"  
}  
],  
"command_type": "CompositeCommand"  
}  
"https://api.qubole.com/api/\${V}/commands"
```

Response:

```
HTTP/1.1 200 OK  
Content-Type: application/json; charset=utf-8
```

```
{  
    "timeout": null,  
    "template": "generic",  
    "resolved_macros": null,  
    "status": "waiting",  
    "qbol_session_id": 974,  
    "progress": 0,  
    "qlog": null,  
    "can_notify": false,  
    "end_time": null,  
    "start_time": null,  
    "user_id": 10,  
    "label": "default",  
    "command": {  
        "sub_commands": [  
            {  
                "status": "waiting",  
                "command": {  
                    "parameters": null,  
                    "archives": null,  
                    "inline": "if hadoop fs -ls /user/hduser/terasort-input; then\n      hadoop fs -rmr /user/hduser/terasort-input\nfi",  
                    "script_location": null,  
                    "files": null  
                },  
                "start_time": null,  
                "end_time": null,  
                "sequence_number": 1,  
                "pid": null,  
                "id": 50,  
                "command_type": "ShellCommand"  
            },  
            {  
                "status": "waiting",  
                "command": {  
                    "parameters": null,  
                    "archives": null,  
                    "inline": "#!/bin/bash\n\nNUM_MAP_TASKS=20\n\n# Total data generated = (DATA_SIZE * 100) kB  
for i in {1..${NUM_MAP_TASKS}}; do  
    dd if=/dev/urandom of=/user/hduser/terasort-input${i} bs=1M count=1000  
done",  
                    "script_location": null,  
                    "files": null  
                },  
                "start_time": null,  
                "end_time": null,  
                "sequence_number": 2,  
                "pid": null,  
                "id": 51,  
                "command_type": "ShellCommand"  
            }  
        ]  
    }  
}
```

```

    "pid": null,
    "id": 51,
    "command_type": "ShellCommand"
},
{
    "status": "waiting",
    "command": {
        "parameters": null,
        "archives": null,
        "inline": "#!/bin/bash\n\nNUM_REDUCE_TASKS=20\nNUM_MAP_TASKS=20\nhadoop jar /usr/lib/hadoop-mapreduce/contrib/stream伪命令.jar -input /user/hive/warehouse/test/ -output /user/hive/warehouse/test/_output -m 1 -r 1 -shards 1 -num-reduce-tasks 20 -num-map-tasks 20",
        "script_location": null,
        "files": null
    },
    "start_time": null,
    "end_time": null,
    "sequence_number": 3,
    "pid": null,
    "id": 52,
    "command_type": "ShellCommand"
}
],
},
"pool": null,
"account_id": 10,
"num_result_dir": -1,
"pid": null,
"created_at": "2015-01-27T14:31:00Z",
"name": null,
"submit_time": 1422369060,
"path": "/tmp/2015-01-27/10/2946",
"id": 2946,
"command_source": "API",
"command_type": "CompositeCommand",
"meta_data": {
    "results_resource": "commands/2946/results",
    "logs_resource": "commands/2946/logs"
}
}
}

```

4.5.19 Submit a Refresh Table Command

This command API can be used to refresh only a Hive table. This API can be mainly used when a Hive partition or directory is extensively used to write data and when Hive tables must be refreshed regularly.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
db_name	Database name that contains the Hive table, which is to be refreshed.
hive_table	Name of the Hive table that is to be refreshed
loader_stable	Checks if a Hive directory or partition is fully loaded.
loader_stable_time	Initial time in minutes to wait before a directory is considered loaded.
template	s3import template is used to refresh tables. The template is used to differentiate the refresh table command from other Hive commands that use generic template.

Request

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d ' {"db_name":"default", "hive_table":"default_qubole_memetracker", "loader_stable":"1", "loader_stable_mult":"Minutes", "template":"s3import"}' "http://api.qubole.com/api/v1.2/commands"
```

Sample Response

```
HTTP/1.1 200 OK
Cache-Control: max-age=0, private, must-revalidate
Content-Type: application/json; charset=utf-8
Date: Mon, 16 Nov 2015 18:49:20 GMT
ETag: "b2a6a723e8b7e931ff87e44feacc9a2f"
Server: nginx/1.6.2 + Phusion Passenger 4.0.53
Set-Cookie: _tapp_session=bd0070f3344489b9a306c8c072cdc71c; path=/; HttpOnly
Set-Cookie: qbol_user_id=1574; path=/
Status: 200 OK
X-Powered-By: Phusion Passenger 4.0.53
X-Rack-Cache: invalidate, pass
X-Request-Id: 91b57e2a21319a4b51fb354378869fb0
X-Runtime: 0.422908
X-UA-Compatible: IE=Edge,chrome=1
Content-Length: 865
Connection: keep-alive

{"status": "waiting", "qbol_session_id": null, "progress": 0, "uid": 2081, "account_id": 632, "end_time": null, "start_time": null, "command_type": "HiveCommand", "command": {"sample": false, "approx_aggregations": false, "query": "use default ; alter table default_qubole_memetracker recover partitions;", "approx_mode": false, "loader_table_name": "default.default_qubole_memetracker", "retry": 0, "script_location": null, "loader_stable": 1, "loader_stable_time": 1, "created_at": "2015-11-16T18:49:20Z", "num_result_dir": 0, "submit_time": 1447699760, "pid": null, "can_notify": true, "qlog": null, "resolved_macros": null, "label": "default", "user_id": 1574, "saved_query mutable_id": null, "command_source": "API", "name": null, "pool": null, "timeout": null, "template": "s3import", "path": "/tmp/2015-11-16/632/402620", "id": 402620, "meta_data": {"results_resource": "commands/402620/results", "logs_resource": "commands/402620/logs"}}
```

4.6 Hive Metadata API

4.6.1 Schema or Database

GET /api/v1.2/hive/default/

Parameters

Parameter	Description
filter	A regular expression to filter the result.
describe	Value has to be <i>true</i> . The result contains columns for each table.

Returns a json array of tables available in Qubole.

Example

Goal

With filter.

```
curl -i -X GET \
-H "Accept: application/json" \
-H "Content-type: application/json" \
-H "X-AUTH-TOKEN: $AUTH_TOKEN" \
"https://api.qubole.com/api/v1.2/hive/default/?filter=.*qubole.*"
```

Sample Response

```
["default_qubole_airline_origin_destination", "default_qubole_memetracker"]
```

Goal

With filter and describe.

```
curl -i -X GET \
-H "Accept: application/json" \
-H "Content-type: application/json" \
-H "X-AUTH-TOKEN: $AUTH_TOKEN" \
"https://api.qubole.com/api/v1.2/hive/default/?filter=.*qubole.*&describe=true"
```

Sample Response

```
[{"columns": [{"name": "break", "type": "string"}, {"name": "coupongeotype", "type": "string"}, {"name": "coupons", "type": "string"}, {"name": "coupontype", "type": "string"}], "name": "default_qubole_airline_origin_destination"}]
```



```
        "type": "string"
},
{
  "name": "opcARRIER",
  "type": "string"
},
{
  "name": "origin",
  "type": "string"
},
{
  "name": "originaptind",
  "type": "string"
},
{
  "name": "origincitynum",
  "type": "string"
},
{
  "name": "origincountry",
  "type": "string"
},
{
  "name": "originstate",
  "type": "string"
},
{
  "name": "originstatefips",
  "type": "string"
},
{
  "name": "originstatename",
  "type": "string"
},
{
  "name": "originwac",
  "type": "string"
},
{
  "name": "passengers",
  "type": "string"
},
{
  "name": "quarter",
  "type": "string"
},
{
  "name": "rpcarrier",
  "type": "string"
},
{
  "name": "seqnum",
  "type": "string"
},
{
  "name": "tkcarrier",
  "type": "string"
},
```

```
{  
    "name": "year",  
    "type": "string"  
},  
]  
}  
,  
{  
    "default_qubole_memetracker": {  
        "columns": [  
            {  
                "name": "lnks",  
                "type": "string"  
            },  
            {  
                "name": "phr",  
                "type": "string"  
            },  
            {  
                "name": "site",  
                "type": "string"  
            },  
            {  
                "name": "ts",  
                "type": "string"  
            },  
            {  
                "name": "month",  
                "type": "string"  
            }  
        ]  
    }  
}  
]
```

4.6.2 Get Table Definition

GET /api/v1.2/hive/default/table****

Use this API to get the table definition.

Example

```
curl -i -X GET -H "Accept: application/json" -H "Content-type: application/json" -H "X-AUTH-TOKEN: $A
```

Sample Response

```
[  
    {  
        "name": "bytes",  
        "type": "int"  
    },  
    {  
        "name": "pagename",  
        "type": "string"  
    }  
]
```

```

    "type": "string"
},
{
  "name": "pageviews",
  "type": "int"
},
{
  "name": "projcode",
  "type": "string"
},
{
  "name": "dt",
  "type": "string"
}
]

```

4.6.3 Store Table Properties

POST /api/v1.2/hive/schema/table

Modify metadata of tables in the given schema of the hive metastore.

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
interval	Number representing the interval at which data is loaded.
interval_unit	Unit of the interval. Valid values are <i>minutes, hours, days, weeks and months</i> .
columns	JSON Hash with Date/Time Format of partition columns. Date format should be a valid input to the <i>strftime</i> function. If there are no partition columns, then it should be an empty hash. For partition columns that are not date/time, the value should be an empty string.

Example

Imagine a table *daily_tick_data* in the *default* hive schema that has the following partitions

1. stock_exchange
2. stock_symbol
3. year
4. date

```

$ cat payload.json
{
  "interval": "1",
  "interval_unit": "days",
  "columns": {
    "stock_exchange": "",
    "stock_symbol": "",
    "year": "%Y",
    "date": "%Y-%m-%d"
  }
}

```

```
}
```

```
$ curl -i -X POST -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "X-AUTH-TOKEN: $AUTH_TOKEN" \
--data @payload.json \
https://api.qubole.com/api/v1.2/hive/default/daily_tick_data/properties
```

Response

```
{"status": "successful"}
```

4.6.4 Get Table Properties

GET /api/v1.2/hive/default/*table*/properties/

Use this API to get the table properties. The column property is an ordered array of partition columns, representing the major partition column first that is followed by the major partition column, and so on.

Example

```
curl -i -X GET -H "Accept: application/json" -H "Content-type: application/json" -H "X-AUTH-TOKEN: $AUTH_TOKEN" https://api.qubole.com/api/v1.2/hive/default/daily_tick_data/properties/
```

Sample Response

```
{
  "columns": [
    {
      "stock_exchange": ""
    },
    {
      "stock_symbol": ""
    },
    {
      "year": "%Y"
    },
    {
      "date1": "%Y-%m-%d"
    }
  ],
  "interval": "1",
  "interval_unit": "days"
}
```

4.6.5 Delete Table Properties

DELETE /api/v1.2/hive/default/ (string: table_name) /properties

Delete properties associated with a Hive table.

Example

Goal

To delete the Hive table properties.

```
curl -i -X DELETE -H "Accept: application/json" \
-H "Content-type: application/json" \
-H "X-AUTH-TOKEN: $AUTH_TOKEN" \
"https://api.qubole.com/api/v1.2/hive/default/daily_tick_data/properties"
```

Response

The response will be a JSON object with success or failure information.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{"status": "successful"}
```

4.7 Cluster API

4.7.1 List all Clusters

GET /api/v1.3/clusters

Get configuration details of all clusters in the account.

Response

The response is an array of hashes. Each hash contains key-value pairs describing various attributes of a cluster.

Example

Goal

To list all the clusters in the account.

```
curl -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN"
-H "Content-Type:application/json" -H "Accept: application/json"
"https://api.qubole.com/api/v1.3/clusters"
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

"security_settings": {
    "encrypted_ephemerals":false
},
```

```
"enable_ganglia_monitoring":false,
"label":[
  "my_cluster"
],
"ec2_settings":{
  "compute_validated":false,
  "compute_secret_key":"<your_ec2_compute_secret_key>",
  "aws_region":"us-west-2",
  "vpc_id":null,
  "aws_preferred_availability_zone":"Any",
  "compute_access_key":"<your_ec2_compute_access_key>",
  "subnet_id":null
},
"node_bootstrap_file":"node_bootstrap.sh",
".hadoop_settings":{
  "use_hadoop2":false,
  "custom_config":null,
  "fairscheduler_settings":{
    "default_pool":null
  }
},
"disallow_cluster_termination":false,
"presto_settings":{
  "enable_presto":false,
  "custom_config":null
},
"id":116,
"state":"DOWN",
"node_configuration":{
  "max_nodes":10,
  "master_instance_type":"m1.large",
  "slave_instance_type":"m1.xlarge",
  "use_stable_spot_nodes":false,
  "slave_request_type":"spot",
  "initial_nodes":1,
  "spot_instance_settings":{
    "maximum_bid_price_percentage":"100.0",
    "timeout_for_request":10,
    "maximum_spot_instance_percentage":60
  }
}
}
```

4.7.2 Create a New Cluster

POST /api/v1.3/clusters/

Creates a new cluster with the given configuration.

Use this to create a new cluster for a workload that has to run in parallel with your pre-existing workloads.

You might want to run workloads across different regions, or on different types of instances, or there could be other reasons for creating a new cluster.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
label	A list of labels that identify the cluster. At least one label must be provided when creating a cluster.
<i>ec2_settings</i>	Amazon EC2 Settings. The default values are considered if the settings are not configured.
<i>node_configuration</i>	Cluster node instances type and other settings.
<i>hadoop_settings</i>	Hadoop cluster settings.
<i>security_settings</i>	Instance security settings.
<i>presto_settings</i>	Presto cluster settings.
<i>disallow_cluster_termination</i>	Prevent auto-termination of the cluster after a prolonged period of disuse. The default value is, <code>false</code> .
<i>enable_ganglia_monitoring</i>	Enable Ganglia monitoring for the cluster. The default value is, <code>false</code> .
<i>node_bootstrap_file</i>	A file that is executed on every node of the cluster at boot time. Use this to customize the cluster nodes by setting up environment variables, installing the required packages, and so on. The default value is, <code>node_bootstrap.sh</code> .

ec2_settings

Parameter	Description
com-pute_access_key	The EC2 Access Key. (<i>Note: This field is not visible to non-admin users.</i>)
com-pute_secret_key	The EC2 Secret Key. (<i>Note: this field is not visible to non-admin users.</i>)
<i>aws_region</i>	The AWS region in which the cluster is created. The default value is, <code>us-east-1</code> . Valid values are, <code>us-east-1</code> , <code>us-west-1</code> , <code>us-west-2</code> , <code>eu-west-1</code> , <code>sa-east-1</code> , <code>ap-southeast-1</code> , and <code>ap-northeast-1</code> .
<i>aws_preferred_availability_zone</i>	The preferred availability zone in which the cluster must be created. The default value is <code>Any</code> .
<i>vpc_id</i>	The ID of the vpc in which the cluster is created. In this vpc, the <code>enableDnsHostnames</code> parameter must be set to true.
<i>subnet_id</i>	The ID of the subnet in which the cluster is created. This subnet must belong to the above vpc. And, it should be a ‘public’ subnet.

node_configuration

Parameter	Description
master_instance_type	The instance type to use for a cluster master node. The default value is m1.large for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.xlarge for a Spark cluster.
slave_instance_type	The instance type to use for cluster slave nodes. The default value is m1.xlarge for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.2xlarge for a Spark cluster.
initial_nodes	The number of nodes to start the cluster with. The default value is 2.
max_nodes	The maximum number of nodes up to which the cluster can be auto-scaled. The default value is 2.
slave_request_type	The purchasing option for the slave instances. The default value is spot. The valid values are, ondemand and spot.
fallback_to_ondemand	Fallback to on-demand nodes if spot nodes could not be obtained. Valid only if slave request type is ‘spot’. The default value is false if slave_request_type is spot.
ebs_volume_type	The default EBS volume type is <i>standard</i> (magnetic). The other possible value is gp2 (ssd). EBS volumes are attached to increase storage on instance types that come with low storage but have good CPU and memory configuration.
ebs_volume_size	The default EBS volume size is 100 GB. The supported value range is 100-500 GB.
ebs_volume_count	The number of EBS volumes to attach to each cluster instance. The default value is 0.
<i>spot_instance_settings</i>	The purchase options for Spot Instances.
<i>stable_spot_instance_settings</i>	Purchases both master node(s) and slave node(s) as Spot Instances only. The bid price is given using the <i>stable_spot_instance_settings</i> .
custom_ec2_tags	<p>It is an optional parameter. Its value contains a <tag> and a <value>. For example, custom-ec2-tags ‘{“key1”:”value1”, “key2”:”value2”}’. A set of tags to be applied on the AWS instances created for the cluster. Specified as a JSON object, for example, {“project”: “webportal”, “owner”: “john@example.com”} It contains a custom tag and value. You can set a custom EC2 tag if you want the instances of a cluster to get that tag on AWS.</p> <p>Tags and values must have alphanumeric characters and can contain only these special characters: + (plus-sign), . (full-stop/period/dot), - (hyphen), @ (at-the-rate of symbol), = (equal sign), / (forward slash), : (colon) and _ (an underscore). The tags, <i>Qubole</i> and <i>alias</i> are reserved for use by Qubole (see <i>Qubole Cluster EC2 Tags</i>). Tags beginning with <i>aws-</i> are reserved for use by Amazon.</p>
use_hadoop2	Set this parameter value to 1 for starting Hadoop-2 daemons on a cluster. It is a mandatory setting for a Hadoop 2 cluster.
use_spark	This is a mandatory setting for a Spark cluster. Its value must be 1 to start Spark daemons on the cluster.

spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the slave node instance type. The default value is 100.
timeout_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.
maxi-mum_spot_instance_percent	The maximum percentage of instances that may be purchased from the AWS Spot market. The default value is 50.

stable_spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the node instance type. The default value is NULL.
time-out_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.

hadoop_settings

Parameter	Description
custom_config	The custom Hadoop configuration overrides. The default value is <i>blank</i> .
fairscheduler_settings	The fair scheduler configuration options.
use_hbase	Starts HBase daemons on the cluster.
use.hadoop2	Set this parameter value to 1 for starting Hadoop-2 daemons on a cluster. It is a mandatory setting for a Hadoop 2 cluster.
use_spark	This is a mandatory setting for a spark cluster. Its value must be 1 to start Spark daemons on the cluster.
use_qubole_placement	Policy Qubole Block Placement policy for clusters with spot nodes.

fairscheduler_settings

Parameter	Description
fairsched-uler_config_xml	The XML string, with custom configuration parameters, for the fair scheduler. The default value is <i>blank</i> .
default_pool	The default pool for the fair scheduler. The default value is <i>blank</i> .

security_settings

Parameter	Description
en-crypted_ephemerals	Encrypt the ephemeral drives on the instance. The default value is, <code>false</code> .
cus-tomer_ssh_key	SSH key to use to login to the instances. The default value is, <code>none</code> . (<i>Note: This field is not visible to non-admin users.</i>) The SSH key must be in the OpenSSH format and not in the PEM/PKCS format.
persistent-security-groups	This option overrides the account-level security group settings. By default, this option is not set but inherits the account-level persistent security group, if any. Use this option if you want to give additional access permissions to cluster nodes.

presto_settings

Parameter	Description
enable_presto	Enables Presto on the cluster.
custom_config	Specifies if the custom Presto configuration overrides. The default value is <i>blank</i> .

Response

The response contains a JSON object representing the created cluster. All the attributes mentioned here are returned (except when otherwise specified or redundant).

Example

Goal

Create a cluster called, ‘my_cluster’ in the ‘us-west-2’ AWS region.

```
curl -X POST -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d {  
    "label": [  
        "my_cluster"  
    ],  
    "node_configuration": {  
        "slave_request_type": "spot",  
        "initial_nodes": 1,  
        "spot_instance_settings": {  
            "timeout_for_request": 10,  
            "maximum_spot_instance_percentage": 60,  
            "maximum_bid_price_percentage": "100.0"  
        },  
        "max_nodes": 10,  
        "master_instance_type": "m1.large",  
        "slave_instance_type": "m1.xlarge"  
    },  
    "hadoop_settings": {  
        "use_hadoop2": false,  
        "max_nodes": 10,  
        "fairscheduler_settings": {  
            "default_pool": null  
        },  
        "custom_config": null  
    },  
    "enable_ganglia_monitoring": false,  
    "state": "DOWN",  
    "node_bootstrap_file": "node_bootstrap.sh",  
    "use_hadoop2": false,  
    "security_settings": {  
        "encrypted_ephemerals": false  
    },  
    "ec2_settings": {  
        "compute_validated": true,  
        "aws_region": "us-west-2",  
        "aws_preferred_availability_zone": "Any",  
        "compute_secret_key": "<your_ec2_compute_secret_key>",  
        "vpc_id": null,  
        "compute_access_key": "<your_ec2_compute_access_key>"  
    }  
}
```

```
    "subnet_id":null
},
"presto_settings":{
    "enable_presto":false,
    "custom_config":null
},
"disallow_cluster_termination":false
}

https://api.qubole.com/api/v1.3/clusters
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
    "security_settings":{
        "encrypted_ephemerals":false
    },
    "enable_ganglia_monitoring":false,
    "label":[
        "my_cluster"
    ],
    "ec2_settings":{
        "compute_validated":false,
        "compute_secret_key":"<your_ec2_compute_secret_key>",
        "aws_region":"us-east-1",
        "vpc_id":null,
        "aws_preferred_availability_zone":"Any",
        "compute_access_key":"<your_ec2_compute_access_key>",
        "subnet_id":null
    },
    "node_bootstrap_file":"node_bootstrap.sh",
    "hadoop_settings":{
        "use_hadoop2":false,
        "custom_config":null,
        "fairscheduler_settings":{
            "default_pool":null
        }
    },
    "disallow_cluster_termination":false,
    "presto_settings":{
        "enable_presto":false,
        "custom_config":null
    },
    "id":116,
    "state":"DOWN",
    "node_configuration":{
        "max_nodes":10,
        "master_instance_type":"m1.large",
        "slave_instance_type":"m1.xlarge",
        "use_stable_spot_nodes":false,
        "slave_request_type":"spot",
        "initial_nodes":1,
        "spot_instance_settings":{
            "maximum_bid_price_percentage":"100.0",
            "timeout_for_request":10,
        }
    }
}
```

```

        "maximum_spot_instance_percentage":60
    }
}
}

```

4.7.3 View Cluster Configuration

GET /api/v1.3/clusters/ (string: id_or_label)

This API is used to get the configuration details of a single cluster. Invoke this API with the cluster ID or a label that is assigned to it.

Example

Get the configuration information of a single cluster

```
curl -X GET -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" https://api.qubole.com/api/v1.3/clusters/1710
```

Response

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
    "security_settings": {
        "encrypted_ephemerals": false
    },
    "enable_ganglia_monitoring": false,
    "label": [
        "my_cluster"
    ],
    "ec2_settings": {
        "compute_validated": false,
        "compute_secret_key": "<your_ec2_compute_secret_key>",
        "aws_region": "us-west-2",
        "vpc_id": null,
        "aws_preferred_availability_zone": "Any",
        "compute_access_key": "<your_ec2_compute_access_key>",
        "subnet_id": null
    },
    "node_bootstrap_file": "node_bootstrap.sh",
    "hadoop_settings": {
        "use_hadoop2": false,
        "custom_config": null,
        "fairscheduler_settings": {
            "default_pool": null
        }
    },
    "disallow_cluster_termination": false,
    "presto_settings": {
        "enable_presto": false,
        "custom_config": null
    }
}

```

```
        },
        "id":116,
        "state":"DOWN",
        "node_configuration":{
            "max_nodes":10,
            "master_instance_type":"m1.large",
            "slave_instance_type":"m1.xlarge",
            "use_stable_spot_nodes":false,
            "slave_request_type":"spot",
            "initial_nodes":1,
            "spot_instance_settings":{
                "maximum_bid_price_percentage":"100.0",
                "timeout_for_request":10,
                "maximum_spot_instance_percentage":60
            }
        }
    }
```

4.7.4 Edit Cluster Configuration

PUT /api/v1.3/clusters/ (*string: id_or_label*)

Edit one or more attributes of an existing cluster.

Most attribute changes takes effect when the cluster is restarted. However, label changes take effect immediately and new commands on such labels are run on the new cluster.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
<i>label</i>	A list of labels that identify the cluster. At least one label must be provided when creating a cluster.
<i>ec2_settings</i>	Amazon EC2 Settings. The default values are considered if the settings are not configured.
<i>node_configuration</i>	Cluster node instances type and other settings.
<i>hadoop_settings</i>	Hadoop cluster settings.
<i>security_settings</i>	Instance security settings.
<i>presto_settings</i>	Presto cluster settings.
<i>disallow_cluster_termination</i>	Prevents auto-termination of the cluster after a prolonged period of disuse. The default value is <code>false</code> .
<i>enable_ganglia_monitoring</i>	Enable Ganglia monitoring for the cluster. The default value is, <code>false</code> .
<i>node_bootstrap_file</i>	A file that gets executed on every node of the cluster at boot time. You can use this to customize your cluster nodes by setting up environment variables, installing required packages, etc. The default value is, <code>node_bootstrap.sh</code> .

ec2_settings

Parameter	Description
compute_access_key	The EC2 Access Key (<i>Note: This field is not visible to non-admin users.</i>)
compute_secret_key	The EC2 Secret Key (<i>Note: This field is not visible to non-admin users.</i>)
aws_region	The AWS region to create the cluster in. The default value is, us-east-1. Valid values are, us-east-1, us-west-1, us-west-2, eu-west-1, sa-east-1, ap-southeast-1, and ap-northeast-1.
aws_preferred_availability_zone	The availability zone in the region to create the cluster in. The default value is Any.
vpc_id	The ID of the VPC to create the cluster in. This VPC must have the enableDnsHostnames variable set to TRUE.
subnet_id	The ID of the subnet where the cluster must be created. This subnet must belong to the above vpc. And it must be a ‘public’ subnet.

node_configuration

Parameter	Description
master_instance_type	The instance type to use for a cluster master node. The default value is m1.large for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.xlarge for a Spark cluster.
slave_instance_type	The instance type to use for cluster slave nodes. The default value is m1.xlarge for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.2xlarge for a Spark cluster.
initial_nodes	The number of nodes to start the cluster with. The default value is 2.
max_nodes	The maximum number of nodes up to which the cluster can be auto-scaled. The default value is 2.
slave_request_type	The purchasing option for the slave instances. The default value is spot. The valid values are, ondemand and spot.
fallback_to_ondemand	Fallback to on-demand nodes if spot nodes could not be obtained. Valid only if slave request type is ‘spot’.The default value is false if slave_request_type is spot.
ebs_volume_type	The default EBS volume type is standard (magnetic). The other possible value is gp2 (ssd). EBS volumes are attached to increase storage on instance types that come with low storage but have good CPU and memory configuration.
ebs_volume_size	The default EBS volume size is 100 GB. The supported value range is 100-500 GB.
ebs_volume_count	It is the number of EBS volumes to attach to each cluster instance. The default value is 0.
<i>spot_instance_settings</i>	The purchase options for Spot Instances.
<i>stable_spot_instance_settings</i>	Purchases both master node(s) and slave node(s) as Spot Instances only. The bid price is given using the <i>stable_spot_instance_settings</i> .
custom_ec2_tags	<p>It is an optional parameter. Its value contains a <tag> and a <value>. For example, custom-ec2-tags ‘{“key1”：“value1”, “key2”：“value2”}’. A set of tags to be applied on the AWS instances created for the cluster. Specified as a JSON object, for example, {“project”: “webportal”, “owner”: “john@example.com”} It contains a custom tag and value. You can set a custom EC2 tag if you want the instances of a cluster to get that tag on AWS.</p> <p>Tags and values must have alphanumeric characters and can contain only these special characters: + (plus-sign), . (full-stop/period/dot), - (hyphen), @ (at-the-rate of symbol), = (equal sign), / (forward slash), : (colon) and _ (an underscore). The tags, <i>Qubole</i> and <i>alias</i> are reserved for use by Qubole (see <i>Qubole Cluster EC2 Tags</i>). Tags beginning with <i>aws-</i> are reserved for use by Amazon.</p>

spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the slave node instance type. The default value is, 100.
timeout_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.
maxi-mum_spot_instance_percent	The maximum percentage of instances that may be purchased from the AWS Spot market. The default value is 50.

stable_spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the node instance type. The default value is NULL.
time-out_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.

hadoop_settings

Parameter	Description
custom_config	The custom Hadoop configuration overrides. The default value is <i>blank</i> .
fairscheduler_settings	The fair scheduler configuration options.
use_hbase	Starts HBase daemons on the cluster.
use.hadoop2	Set this parameter value to 1 for starting Hadoop-2 daemons on a cluster. It is a mandatory setting for a Hadoop 2 cluster.
use_spark	This is a mandatory setting for a spark cluster. Its value must be 1 to start Spark daemons on the cluster.
use_qubole_placement	Policy Qubole Block Placement policy for clusters with spot nodes.

fairscheduler_settings

Parameter	Description
fairsched-uler_config_xml	XML string with custom configuration parameters for the fair scheduler. The default value is, <i>blank</i> .
default_pool	The default pool for the fair scheduler. The default value is, <i>blank</i> .

security_settings

Parameter	Description
en-crypted_ephemerals	Encrypt the ephemeral drives on the instance. The default value is, <i>false</i> .
cus-tomer_ssh_key	SSH key to use to login to the instances. The default value is, <i>none</i> . (<i>Note: This field is not visible to non-admin users.</i>) The SSH key must be in the OpenSSH format and not in the PEM/PKCS format.
persistent-security-groups	This option overrides the account-level security group settings. By default, this option is not set but inherits the account-level persistent security group, if any. Use this option if you want to give additional access permissions to cluster nodes.

presto_settings

Parameter	Description
enable_presto	Enable Presto on the cluster.
custom_config	Custom Presto configuration overrides. The default is, <i>blank</i> .

Response

The response contains a JSON object representing the edited cluster. All the attributes mentioned here are returned (except when otherwise specified or redundant).

Examples**Goal**

Modify some attributes of the cluster with a cluster ID, 116.

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d '{
  "cluster": {
    "node_configuration": {
      "initial_nodes": 2,
      "slave_request_type": "ondemand",
      "slave_instance_type": "c3.xlarge",
      "max_nodes": 12,
      "master_instance_type": "c3.large"
    },
    "enable_ganglia_monitoring": true
  }
}'
https://api.qubole.com/api/v1.3/clusters/116
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "security_settings": {
    "encrypted_ephemerals": false
  },
  "enable_ganglia_monitoring": true,
  "label": [
    "my_cluster"
  ],
  "ec2_settings": {
    "compute_validated": false,
    "compute_secret_key": "<your_ec2_compute_secret_key>",
    "aws_region": "us-west-2",
    "vpc_id": null,
    "aws_preferred_availability_zone": "Any",
    "compute_access_key": "<your_ec2_compute_access_key>",
    "subnet_id": null
  },
  "node_bootstrap_file": "node_bootstrap.sh",
  "hadoop_settings": {}
```

```

"use_hadoop2":false,
"custom_config":null,
"fairscheduler_settings":{
    "default_pool":null
},
},
"disallow_cluster_termination":false,
"presto_settings":{
"enable_presto":false,
"custom_config":null
},
"id":116,
"state":"DOWN",
"node_configuration":{
"max_nodes":12,
"master_instance_type":"c3.large",
"slave_instance_type":"c3.xlarge",
"use_stable_spot_nodes":false,
"slave_request_type":"ondemand",
"initial_nodes":2,
"spot_instance_settings":{
"maximum_bid_price_percentage":"100.0",
"timeout_for_request":10,
"maximum_spot_instance_percentage":60
}
}
}
}

```

4.7.5 Clone a Cluster

POST /api/v1.3/clusters/ (string: id_or_label) /clone

Clone a cluster from an existing one. All the attributes of the source cluster (except the label) are copied over to the new cluster, but you can override any of them when creating the clone.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and use the values from the source cluster.

Parameter	Description
<code>label</code>	A list of labels that identify the cluster. At least one label must be provided when creating a cluster. You must provide a new label to clone a cluster.
<code>ec2_settings</code>	Amazon EC2 Settings. The default values are considered if the settings are not configured.
<code>node_configuration</code>	Cluster node instances type and other settings.
<code>hadoop_settings</code>	Hadoop cluster settings.
<code>security_settings</code>	Instance security settings.
<code>presto_settings</code>	Presto cluster settings.
<code>disallow_cluster_termination</code>	Prevents auto-termination of the cluster after a prolonged period of disuse. The default value is <code>false</code> .
<code>enable_ganglia_monitoring</code>	Enable Ganglia monitoring for the cluster. The default value is, <code>false</code> .
<code>node_bootstrap_file</code>	A file that gets executed on every node of the cluster at boot time. You can use this to customize your cluster nodes by setting up environment variables, installing required packages, etc. The default value is, <code>node_bootstrap.sh</code> .

ec2_settings

Parameter	Description
<code>compute_access_key</code>	The EC2 Access Key (<i>Note: This field is not visible to non-admin users.</i>)
<code>compute_secret_key</code>	The EC2 Secret Key (<i>Note: This field is not visible to non-admin users.</i>)
<code>aws_region</code>	The AWS region to create the cluster in. The default value is, <code>us-east-1</code> . Valid values are, <code>us-east-1</code> , <code>us-west-1</code> , <code>us-west-2</code> , <code>eu-west-1</code> , <code>sa-east-1</code> , <code>ap-southeast-1</code> , and <code>ap-northeast-1</code> .
<code>aws_preferred_availability_zone</code>	The availability zone in the region to create the cluster in. The default value is Any.
<code>vpc_id</code>	The ID of the VPC to create the cluster in. This VPC must have the <code>enableDnsHostnames</code> variable set to TRUE.
<code>subnet_id</code>	The ID of the subnet where the cluster must be created. This subnet must belong to the above vpc. And it must be a ‘public’ subnet.

node_configuration

Parameter	Description
master_instance_type	The instance type to use for a cluster master node. The default value is m1.large for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.xlarge for a Spark cluster.
slave_instance_type	The instance type to use for cluster slave nodes. The default value is m1.xlarge for Hadoop-1, Hadoop-2, and Presto clusters. The default value is m3.2xlarge for a Spark cluster.
initial_nodes	The number of nodes to start the cluster with. The default value is 2.
max_nodes	The maximum number of nodes up to which the cluster can be auto-scaled. The default value is 2.
slave_request_type	The purchasing option for the slave instances. The default value is spot. The valid values are, ondemand and spot.
fallback_to_ondemand	Fallback to on-demand nodes if spot nodes could not be obtained. Valid only if slave request type is ‘spot’. The default value is false if slave_request_type is spot.
ebs_volume_type	The default EBS volume type is <i>standard</i> (magnetic). The other possible value is gp2 (ssd). EBS volumes are attached to increase storage on instance types that come with low storage but have good CPU and memory configuration.
ebs_volume_size	The default EBS volume size is 100 GB. The supported value range is 100-500 GB.
ebs_volume_count	The number of EBS volumes to attach to each cluster instance. The default value is 0.
custom_ec2_tags	<p>It is an optional parameter. Its value contains a <tag> and a <value>. For example, custom-ec2-tags ‘{“key1”:“value1”, “key2”:“value2”}’. A set of tags to be applied on the AWS instances created for the cluster. Specified as a JSON object, for example,{“project”: “webportal”, “owner”: “john@example.com”} It contains a custom tag and value. You can set a custom EC2 tag if you want the instances of a cluster to get that tag on AWS.</p> <p>Tags and values must have alphanumeric characters and can contain only these special characters: + (plus-sign), . (full-stop/period/dot), - (hyphen), @ (at-the-rate of symbol), = (equal sign), / (forward slash), : (colon) and _ (an underscore). The tags, <i>Qubole</i> and <i>alias</i> are reserved for use by Qubole (see <i>Qubole Cluster EC2 Tags</i>). Tags beginning with <i>aws-</i> are reserved for use by Amazon.</p>
spot_instance_settings	The purchase options for Spot Instances.
stable_spot_instance_settings	Purchases both master node(s) and slave node(s) as Spot Instances only. The bid price is given using the <i>stable_spot_instance_settings</i> .

spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the slave node instance type. The default value is 100.
timeout_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.
maxi-mum_spot_instance_percent	The maximum percentage of instances that may be purchased from the AWS Spot market. The default value is 50.

stable_spot_instance_settings

Parameter	Description
maxi-mum_bid_price_percent	The maximum value to bid for Spot Instances. This is expressed as a percentage of the base price for the node instance type. The default value is NULL.
time-out_for_request	The timeout for a Spot Instance request in minutes. The default value is 10.

hadoop_settings

Parameter	Description
custom_config	The custom Hadoop configuration overrides. The default value is <i>blank</i> .
fairscheduler_settings	The fair scheduler configuration options.
use_hbase	Starts HBase daemons on the cluster.
use.hadoop2	Set this parameter value to 1 for starting Hadoop-2 daemons on a cluster. It is a mandatory setting for a Hadoop 2 cluster.
use_spark	This is a mandatory setting for a spark cluster. Its value must be 1 to start Spark daemons on the cluster.
use_qubole_placement	Policy Qubole Block Placement policy for clusters with spot nodes.

fairscheduler_settings

Parameter	Description
fairsched-uler_config_xml	XML string with custom configuration parameters for the fair scheduler. The default value is, <i>blank</i> .
default_pool	The default pool for the fair scheduler. The default value is, <i>blank</i> .

security_settings

Parameter	Description
en-crypted_ephemerals	Encrypt the ephemeral drives on the instance. The default value is, <i>false</i> .
cus-tomer_ssh_key	SSH key to use to login to the instances. The default value is, <i>none</i> . (<i>Note: This field is not visible to non-admin users.</i>)
persistent-security-groups	This option overrides the account-level security group settings. By default, this option is not set but inherits the account-level persistent security group, if any. Use this option if you want to give additional access permissions to cluster nodes.

presto_settings

Parameter	Description
enable_presto	Enable Presto on the cluster.
custom_config	Custom Presto configuration overrides. The default is, <i>blank</i> .

Response

The response contains a JSON object representing the new cluster. All the attributes mentioned here are returned (except when otherwise specified or redundant).

Examples

Goal

Clone the cluster with ID 116.

```
curl -X POST -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d '{
  "cluster": {
    "label": ["116_clone"],
    "node_configuration": {
      "initial_nodes": 2,
      "slave_request_type": "ondemand",
      "slave_instance_type": "c3.xlarge",
      "max_nodes": 10,
      "master_instance_type": "c3.large"
    },
    "enable_ganglia_monitoring": true
  }
}' \
https://api.qubole.com/api/v1.3/clusters/116/clone
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "security_settings": {
    "encrypted_ephemerals": false
  },
  "enable_ganglia_monitoring": true,
  "label": [
    "116_clone"
  ],
  "ec2_settings": {
    "compute_validated": false,
    "compute_secret_key": "<your_ec2_compute_secret_key>",
    "aws_region": "us-west-2",
    "vpc_id": null,
    "aws_preferred_availability_zone": "Any",
    "compute_access_key": "<your_ec2_compute_access_key>",
    "subnet_id": null
  },
  "node_bootstrap_file": "node_bootstrap.sh",
  "node_bootstrap_command": "sh node_bootstrap.sh"
}
```

```
"hadoop_settings":{  
  "use_hadoop2":false,  
  "custom_config":null,  
  "fairscheduler_settings":{  
    "default_pool":null  
  }  
},  
"disallow_cluster_termination":false,  
"presto_settings":{  
  "enable_presto":false,  
  "custom_config":null  
},  
"id":116,  
"state":"DOWN",  
"node_configuration":{  
  "max_nodes":10,  
  "master_instance_type":"c3.large",  
  "slave_instance_type":"c3.xlarge",  
  "use_stable_spot_nodes":false,  
  "slave_request_type":"ondemand",  
  "initial_nodes":2,  
  "spot_instance_settings":{  
    "maximum_bid_price_percentage":"100.0",  
    "timeout_for_request":10,  
    "maximum_spot_instance_percentage":60  
  }  
}  
}
```

4.7.6 Start or Terminate a Cluster

PUT /api/v1.3/clusters/ (string: id_or_label) /state

This API is used to start or terminate a cluster.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
State	Action to perform – valid values are <i>start</i> to start a cluster and <i>terminate</i> to terminate it. Starting a running cluster or stopping a terminated cluster will have no effect.

Examples

Start a cluster

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"state":"start"}' https://api.qubole.com/api/v1.3/clusters/1710/state
```

Response

```
{ "message": "Starting cluster with id 1710." }
```

Terminate a running cluster

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"state":"terminate"}' https://api.qubole.com/api/v1.3/clusters/1710/state
```

Response

```
{ "message": "Terminating cluster with id 1710." }
```

4.7.7 Check Cluster Status

GET /api/v1.3/clusters/ (string: id_or_label)

This API is used to get the current state of a cluster.

Examples**Get the state of a running cluster**

```
curl -X GET -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" https://api.qubole.com/api/v1.3/clusters/1710/state
```

Response

```
{
  "state": "UP",
  "last_health_check_action_at": null,
  "start_at": "2014-04-25T14:08:28Z",
  "down_at": null,
  "cluster_id": 1710,
  "terminate_reason": null,
  "nodes": [
    {
      "instance_type": "m1.large",
      "ec2_instance_id": "i-abcdefdd",
      "is_spot_instance": false,
      "up_time": "2014-04-25T14:06:02Z",
      "private_ip": "domU-11-22-33-44-55-02.compute-1.internal",
      "last_seen_time": "2014-04-25T14:06:02Z",
      "role": "master",
      "down_time": null
    }
  ]
}
```

```
        "hostname": "ec2-55-66-33-120.compute-1.amazonaws.com"
    },
    {
        "instance_type": "m1.xlarge",
        "ec2_instance_id": "i-abcdef54",
        "is_spot_instance": false,
        "up_time": "2014-04-25T14:06:02Z",
        "private_ip": "ip-11-22-33-44.ec2.internal",
        "last_seen_time": "2014-04-25T14:06:02Z",
        "role": "node0001",
        "down_time": null,
        "hostname": "ec2-55-88-44-120.compute-1.amazonaws.com"
    },
    {
        "instance_type": "m1.xlarge",
        "ec2_instance_id": "i-abcdef55",
        "is_spot_instance": false,
        "up_time": "2014-04-25T14:06:02Z",
        "private_ip": "ip-11-66-111-222.ec2.internal",
        "last_seen_time": "2014-04-25T14:06:02Z",
        "role": "node0002",
        "down_time": null,
        "hostname": "ec2-55-222-111-100.compute-1.amazonaws.com"
    }
],
"last_health_check_action": null,
"config_id": 2196
}
```

Get the state of a terminated cluster

```
curl -X GET -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" https://api.qubole.com/api/v1.3/clusters/1711/state
```

Response

```
{"state": "DOWN"}
```

4.7.8 Delete a Cluster

```
DELETE /api/v1.3/clusters/
```

This API is used to delete a cluster.

Note: The cluster that has the label as ‘default’ cannot be deleted. Hence, there is always at least one cluster in an account.

Examples

Goal

Delete a cluster.

```
curl -X DELETE
-H "X-AUTH-TOKEN:$AUTH_TOKEN"
-H "Content-Type: application/json" -H "Accept: application/json"
https://api.qubole.com/api/v1.3/clusters/1710
```

Response

Returns a message with the status of the delete operation.

```
{“message”：“Successfully deleted cluster with id 1710.”}
```

4.7.9 Reassign Cluster Label

PUT /api/v1.3/clusters/reassign-label

This API is used to reassign a label from one cluster to another.

Parameters

Parameter	Description
label	Label to be moved <i>from</i> the source cluster.
destination_cluster	ID or label of the cluster to move the label <i>to</i> .

Examples

Reassign a label from one cluster to another

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
-d '{"label": "pig_workflow", "destination_cluster": 1710}' https://api.qubole.com/api/v1.3/clusters/reassign-label
```

Response

```
{“message”：“Reassigned cluster label pig_workflow from cluster with id 1711 to cluster with id 1710.”}
```

4.7.10 Run Adhoc Scripts on a Cluster

To run an adhoc script, you can use a REST API to execute a script located in S3 on the cluster.

REST API Endpoint Details

- **RequestType:** PUT
- **RequestEndpoint:** <https://api.qubole.com/api/v1.3/clusters/9545/runscript.json/>
- **Request Parameters:** script:{location of s3 path}

Note: The adhoc script is executed as the root user on the cluster.

You can use the Curl utility to spawn adhoc scripts. You can use the following command to execute the script.

```
curl -X PUT -H "X-AUTH-TOKEN: $Auth-token" -H "Content-Type: application/json" -H "Accept: application/json"
```

Example:

```
curl -X PUT -H "X-AUTH-TOKEN: xxXXXXXX" -H "Content-Type: application/json" -H "Accept: application/json"
```

After the script execution (provided as an example), the following message is displayed:

```
Successfully spawned script on the cluster, Please check the logs for each node at: /media/ephemeral0/logs/
```

Check the log location in the Web node and Cluster (Master/Slave) nodes for error log messages in case of any cluster issue.

The log from each machine is uploaded into the S3 directory using the Cron utility.

Note that every execution of this API creates a new GUID, which can be used to distinguish between the differently executing instances on the API in the cluster. The corresponding logs are being located in the GUID directory inside the `/media/ephemeral0/logs/` directory.

4.7.11 Cluster Metrics

GET /api/v1.3/clusters/(string: id_or_label) /metrics

Note: The metrics are available for clusters running with Ganglia monitoring enabled.

Parameters

Parameter	Description
metric	The metric to monitor. It is possible to get metric values for a particular node or aggregated across cluster
interval	The interval for which the metric values are required. Valid value for <i>interval</i> can be hour, 2hr, 4hr, day, week, month or year. Default interval value is hour.
host-name	The hostname for which the metric values are required. Valid value is the private DNS name of the host. See Per-host Metrics below. If not specified, for certain metrics, API returns the metric value aggregated across the cluster. See Aggregate Cluster Metrics below.

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Per-host Metrics

Metrics related to a host can be collected with *hostname* parameter value specified as the internal DNS name of the instance (with format ip-A-B-C-D.ec2.internal). Some of the useful metrics are:

System Metrics

- *cpu_user* : Percentage of CPU utilization while executing at the user level
- *cpu_system* : Percentage of CPU utilization while executing at the system level

- *cpu_idle* : Percentage of time CPU were idle
- *disk_free* : Total free disk space
- *mem_free* : Amount of available memory
- *bytes_in* : Number of bytes in per second
- *bytes_out* : Number of bytes out per second

Hadoop 1 JobTracker Metrics

Various metrics related to JobTracker can be queried with *hostname* parameter set to `master`.

- Metrics for Hadoop jobs
 - *mapred.jobtracker.jobs_submitted* : Number of Hadoop jobs submitted
 - *mapred.jobtracker.jobs_running* : Number of Hadoop jobs running
 - *mapred.jobtracker.jobs_completed* : Number of Hadoop jobs completed
 - *mapred.jobtracker.jobs_failed* : Number of Hadoop jobs failed
- Metrics for Hadoop map tasks
 - *mapred.jobtracker.map_slots* : Number of map slots
 - *mapred.jobtracker.occupied_map_slots* : Number of map slots occupied
 - *mapred.jobtracker.maps_launched* : Number of map tasks launched
 - *mapred.jobtracker.running_maps* : Number of running map tasks
 - *mapred.jobtracker.waiting_maps* : Number of waiting map tasks
 - *mapred.jobtracker.maps_completed* : Number of map tasks completed
 - *mapred.jobtracker.maps_failed* : Number of map tasks failed
- Metrics for Hadoop reduce tasks
 - *mapred.jobtracker.reduce_slots* : Number of reduce slots
 - *mapred.jobtracker.occupied_reduce_slots* : Number of reduce slots occupied
 - *mapred.jobtracker.reduces_launched* : Number of reduce tasks launched
 - *mapred.jobtracker.running_reduces* : Number of running reduce tasks
 - *mapred.jobtracker.waiting_reduces* : Number of waiting reduce tasks
 - *mapred.jobtracker.reduces_completed* : Number of reduce tasks completed
 - *mapred.jobtracker.reduces_failed* : Number of reduce tasks failed

Examples

The following curl command reports the number of maps launched over the past one hour.

```
curl -i -H "X-AUTH-TOKEN: ${X_AUTH_TOKEN}" -H "Content-Type: application/json" -H "Accept: application/json" -G -d metric=mapred.jobtracker.maps_launched -d hostname=master -d interval=hour https://api.qubole.com/api/v1.3/clusters/${CLUSTER_ID}/metrics
```

The JSON response to the API call contains *datapoints* corresponding to the metric values. The value pair in *datapoints* has the format [metric value, time represented in epoch seconds]. Metric value of “NaN” refers to an unavailable value at that point of time.

Response:

```
[  
  {  
    "datapoints": [  
      [14313937,1427752170],  
      [14313937,1427752185],  
      [14319826.6,1427752200],  
      [14328661,1427752215],  
      ...  
      [14940674,1427755710],  
      [14943716,1427755725],  
      ["NaN",1427755740],  
      ["NaN",1427755755]  
    ],  
    "hostname": "master",  
    "metric": "master last hour",  
    "interval": "hour"  
  }  
]
```

Aggregate Cluster Metrics

Some of the system metrics can be aggregated across cluster to get a broader view of the resource across all instances in the cluster. The *hostname* parameter should not be specified for aggregate cluster metrics.

Some of the useful aggregate cluster metrics are:

- *cpu_report* : Aggregate report of CPU utilization percentage
- *mem_report* : Aggregate report of memory usage in bytes
- *load_report* : Aggregate report with current load, number of processes running processes, nodes and CPU count
- *network_report*: Aggregate report with network traffic in and out of the cluster nodes

Example

```
curl -i -H "X-AUTH-TOKEN: ${X_AUTH_TOKEN}" -H "Content-Type: application/json" -H "Accept: application/json"  
-G  
-d metric=cpu_report  
-d interval=hour  
https://api.qubole.com/api/v1.3/clusters/${CLUSTER_ID}/metrics
```

Response:

```
[  
  {"metric": "User\\g", "interval": "hour", "datapoints": [[[58.689508632,1427752170],[57.445152722,1427752170],...]  
  {"metric": "Nice\\g", "interval": "hour", "datapoints": [[[0.010491367862,1427752170],[0.0088977423639,1427752170],...]  
  {"metric": "System\\g", "interval": "hour", "datapoints": [[[6.4996015936,1427752170],[6.3784860558,1427752170],...]  
  {"metric": "Wait\\g", "interval": "hour", "datapoints": [[[0.44156706507,1427752170],[0.45962815405,1427752170],...]  
  {"metric": "Steal\\g", "interval": "hour", "datapoints": [[[0.096812749004,1427752170],[0.096679946879,1427752170],...]  
  {"metric": "Idle\\g", "interval": "hour", "datapoints": [[[34.283532537,1427752170],[35.6333333333,1427752170],...]  
]
```

4.7.12 Resize a Cluster

PUT /api/v1.3/clusters/ (string: *id_or_label*)

Use this REST API to change the minimum and maximum size of a running cluster.

Note: Set the configuration option, `push` to `true` for changes to be effective immediately without a cluster restart.

Modifying the Minimum Cluster Size

Any hadoop cluster running on Qubole has a minimum and maximum cluster size between which the cluster autoscales depending on the load. The minimum and maximum cluster size can be changed for a running cluster as a pushable configuration.

Request API Template

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"node_configuration": {"initial_nodes": "<value>"}, "push": "true"}' "https://api.qubole.com/api/v1.3/clusters/${cluster_id}"/"
```

Parameters

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Parameter	Description
cluster_id	It is the cluster ID.
node_configuration	This contains the cluster nodes details. See <i>Node Configuration Options used to Minimize Cluster Nodes</i> .

Sample Request

This sample request is for minimizing the nodes for a running cluster with ID 3711.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"node_configuration": {"initial_nodes": "2"}, "push": "true"}' "https://api.qubole.com/api/v1.3/clusters/3711"/"
```

Node Configuration Options used to Minimize Cluster Nodes

Parameter	Description
initial_nodes	The number of minimum cluster nodes in a cluster.
push	Set this parameter to <code>true</code> to change the minimum number of nodes for a running cluster. If <code>push</code> is not set to <code>true</code> , changes become effective only after the cluster is restarted.

Modifying the Maximum Cluster Size

This API is used to modify the maximum number of cluster nodes in a running cluster.

Request API Template

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"node_configuration": {"max_nodes": "<value>"}, "push": "true"}' "https://api.qubole.com/api/v1.3/clusters/${cluster_id}/"
```

Parameters

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Parameter	Description
cluster_id	It is the cluster ID.
node_configuration	This contains the cluster nodes details. See <i>Node Configuration Options used to Maximize Cluster Nodes</i> .

Node Configuration Options used to Maximize Cluster Nodes

Parameter	Description
max_nodes	Use this parameter to change the maximum number of cluster nodes.
push	Set this parameter to <code>true</code> to change the maximum number of nodes for a running cluster. If <code>push</code> is not set to <code>true</code> , changes become effective only after the cluster is restarted.

Sample Request

This sample request is for maximizing the nodes for a running cluster with ID 3850.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d '{"node_configuration": {"max_nodes": "1000"}, "push": "true"}' "https://api.qubole.com/api/v1.3/clusters/3850/"
```

4.7.13 Add a Node to a Cluster

POST /api/v1.3/clusters/ (string: id_or_label) /nodes

This API adds a new slave node to the cluster. The instance type is picked up from the cluster configuration. This action starts the operation asynchronously. The operation can be monitored using the command ID in the response through the *command status* API.

Note: A cluster must be running to add a new node. An Add Node API does not check for maximum size of the cluster. Currently, this function is supported for HBase and Hadoop-1 clusters only.

See [Manually Scaling a Hadoop Cluster](#) for more information.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" '{"node_count": "$value"}' "https://api.qubole.com/api/latest/clusters/${cluster_id}/nodes"
```

Parameters

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Parameter	Description
cluster_id	Hadoop Cluster ID
node_count	The total number of nodes. The default value of node count is 1. (Only hadoop1)

Response

The response contains a JSON object representing the command ID of the add node operation. All attributes mentioned here are returned (except when otherwise specified or redundant).

Example

Request

```
curl -X POST -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" https://api.qubole.com//api/v1.3/clusters/353181/nodes
```

Note: The response is the same as a query_hist response.

Response

```
{
  "submit_time": 1431419457,
  "status": "waiting",
  "saved_query Mutable_id": null,
  "command_source": "",
  "progress": 0,
  "resolved_macros": null,
  "end_time": null,
  "start_time": null,
  "command": {
    "cluster_inst_id": 17991,
    "parameters": null,
    "action": "add",
    "private_dns": null
  },
  "pid": null,
  "qbol_session_id": 80475,
  "command_type": "ClusterManageCommand",
  "created_at": "2015-05-12T08:30:57Z",
  "account_id": 14,
  "name": null,
  "timeout": null,
  "template": "generic",
}
```

```
"label": "Hadoop",
"user_id": 1432,
"pool": null,
"num_result_dir": -1,
"qlog": null,
"path": "/tmp/2015-05-12/14/353181",
"id": 353181,
"meta_data": {
    "results_resource": "commands/353181/results",
    "logs_resource": "commands/353181/logs"
},
"can_notify": false
}
```

4.7.14 Replace a Node in a Cluster

PUT /api/v1.3/clusters/(string: id_or_label)/nodes

This API replaces a slave node in a cluster. The instance type is picked up from the cluster configuration. This action starts the operation asynchronously. The operation can be monitored using the command ID in the response through the *command status* API.

Note: A cluster must be running to replace a node. A Replace Node API does not check for maximum size of the cluster. Currently, this function is supported for HBase clusters only.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
command	The only allowed value is <code>replace</code> .
private_dns	Private DNS of the slave node.

Response

The response contains a JSON object representing the command ID of the replace node operation. All attributes mentioned here are returned (except when otherwise specified or redundant).

Example: Replacing a Node in a Cluster with ID 27623

Request

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d { "private_dns" : "ip-10-234-18-139.ec2.internal", "command" : "replace" } "https://api.qubole.com/api/latest/clusters/${cluster_id}/nodes"
```

Response

```
{
    "command_source": "",
    "pid": null,
    "qlog": null,
    "command_type": "ClusterManageCommand",
    "id": 277344,
    "saved_query mutable_id": null,
    "user_id": 3635,
    "label": "hbase_opentsdb2",
    "template": "generic",
    "qbol_session_id": 40375,
    "progress": 0,
    "can_notify": false,
    "status": "waiting",
    "account_id": 1612,
    "start_time": null,
    "meta_data": {
        "results_resource": "commands/277344/results",
        "logs_resource": "commands/277344/logs"
    },
    "num_result_dir": -1,
    "submit_time": 1432291974,
    "path": "/tmp/2015-05-22/1612/277344",
    "pool": null,
    "resolved_macros": null,
    "name": null,
    "created_at": "2015-05-22T10:52:54Z",
    "command": {
        "action": "replace",
        "private_dns": "ip-10-234-18-139.ec2.internal",
        "cluster_inst_id": 23528,
        "parameters": null
    },
    "end_time": null,
    "timeout": null
}
}
```

4.7.15 Remove a Node from a Cluster

DELETE /api/v1.3/clusters/ (string: id_or_label) /nodes

This API removes a slave node from a cluster. This action starts the operation asynchronously. The operation can be monitored using the command ID in the response through the [command status](#) API.

Note: A cluster must be running to remove a node. A Remove Node API does not check for maximum size of the cluster. Currently, this function is supported for HBase clusters only.

Response

The response contains a JSON object representing the command ID of the remove node operation. All the attributes mentioned here are returned (except when otherwise specified or redundant).

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
private_dns	Private DNS of the slave node.

Example

```
curl -X DELETE  
-H "X-AUTH-TOKEN:$AUTH_TOKEN"  
-H "Content-Type:application/json"  
https://api.qubole.com/api/v1.3/clusters/277344/nodes
```

Response

```
{  
    "command_source": "",  
    "pid": null,  
    "qlog": null,  
    "command_type": "ClusterManageCommand",  
    "id": 277344,  
    "saved_query Mutable_id": null,  
    "user_id": 3635,  
    "label": "hbase_opentsdb2",  
    "template": "generic",  
    "qbol_session_id": 40375,  
    "progress": 0,  
    "can_notify": false,  
    "status": "waiting",  
    "account_id": 1612,  
    "start_time": null,  
    "meta_data": {  
        "results_resource": "commands/277344/results",  
        "logs_resource": "commands/277344/logs"  
    },  
    "num_result_dir": -1,  
    "submit_time": 1432291974,  
    "path": "/tmp/2015-05-22/1612/277344",  
    "pool": null,  
    "resolved_macros": null,  
    "name": null,  
    "created_at": "2015-05-22T10:52:54Z",  
    "command": {  
        "action": "remove",  
        "private_dns": "ip-10-234-18-139.ec2.internal",  
        "cluster_inst_id": 23528,  
        "parameters": null  
    },  
    "end_time": null,  
    "timeout": null  
}
```

4.7.16 Take an HBase Snapshot

POST /api/v1.3/clusters/ (string: id_or_label) /snapshot

A snapshot encapsulates metadata and data in the cluster at a given point in time. It lets an administrator to get back to a previous state of the cluster. An HBase snapshot lets you to take a snapshot of table data with a minimum impact on the Region Servers.

Note: A cluster must be running to take a snapshot. Currently, this function is supported for HBase clusters only.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
s3_location	Private DNS of the slave node.
type	Denotes the type of backup. The default value is full . The other value is incremental that is the difference of previous backup and current data.

Example: Taking an HBase Snapshot from a Cluster with ID 10929

Request

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" -d { "s3_location": "s3://hbasedata.com/opentsdb_backup_22_5_2015" } "https://api.qubole.com/api/latest/clusters/${cluster_id}/snapshots"
```

Response

```
{
  "command_source": "",
  "pid": null,
  "qlog": null,
  "command_type": "ClusterManageCommand",
  "id": 356878,
  "saved_query mutable_id": null,
  "user_id": 1432,
  "label": "hbase_opentsdb2",
  "template": "generic",
  "qbol_session_id": 81801,
  "progress": 0,
  "can_notify": false,
  "status": "waiting",
  "account_id": 14,
  "start_time": null,
  "meta_data": {
    "results_resource": "commands/356878/results",
    "logs_resource": "commands/356878/logs"
  },
  "num_result_dir": -1,
  "submit_time": 1432284955,
  "path": "/tmp/2015-05-22/14/356878",
```

```
"pool": null,
"resolved_macros": null,
"name": null,
"created_at": "2015-05-22T08:55:55Z",
"command": {
    "action": "snapshot",
    "md_cmd": true,
    "private_dns": null,
    "cluster_inst_id": 17991,
    "parameters": {
        "backup_type": "full",
        "s3_location": "s3://hbasedata.com/opentsdb_backup_22_5_2015"
    }
},
"end_time": null,
"timeout": null
}
```

4.7.17 View an HBase Snapshot Schedule

GET /api/v1.3/clusters/(string: id_or_label) /snapshot_schedule

This API returns information on status, frequency, and backup location of an HBase snapshot schedule.

Note: A cluster must be running to view an HBase snapshot table.

Example

Request

```
curl -X GET -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d /api/v1.3/clusters/hbasecluster/snapshot_schedule
```

Response

```
{"status": "SUSPENDED", "frequency_num": 1, "frequency_unit": "days", "id": 205, "s3_location": "s3://location"}
```

4.7.18 Restore HBase Tables

PUT /api/v1.3/clusters/(string: id_or_label) /restore_point

This API is used to restore the HBase tables to the time when a snapshot was taken. Whenever a full/incremental data backup is done, a snapshot is created and a backup ID is generated. Hbase tables can be restored to that point using the backup (snapshot) ID.

Note: A cluster must be running to restore an HBase snapshot table.

Parameters

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Parameter	Description
auto-matic	With this option, dependencies are automatically restored together with the to-be-restored backup/snapshot. Without this option, only the backup/snapshot associated with the specified backup ID is restored. The default value is set to true.
backup_id	A backup ID that is generated whenever a full/incremental data backup is done or for each snapshot.
over-write	When this option is used, running the API overwrites the existing table (if there is any) in the restore target. The default value is set to true.
s3_location	The S3 location that contains the snapshot.
ta-ble_names	Denotes the table(s) to which the snapshot data will be restored. It is mandatory to specify a table name. A table name can be a single table name or comma-separated table names.

Example

Request

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d { "table_names": "tsdb", "backup_id": "backup_1432284975205", "automatic": true, "overwrite": true, "s3_location": "s3://hbasetabledata.com/opentsdb_backup_22_5_2015" } "https://api.qubole.com/api/latest/clusters/${cluster_id}/restore_point"
```

Response

```
{
  "command_source": "",
  "pid": null,
  "qlog": null,
  "command_type": "ClusterManageCommand",
  "id": 277336,
  "saved_query mutable_id": null,
  "user_id": 3635,
  "label": "opentsdb2",
  "template": "generic",
  "qbol_session_id": 40375,
  "progress": 0,
  "can_notify": false,
  "status": "waiting",
  "account_id": 1612,
  "start_time": null,
  "meta_data": {
    "results_resource": "commands/277336/results",
    "logs_resource": "commands/277336/logs"
  },
  "num_result_dir": -1,
  "submit_time": 1432290500,
  "path": "/tmp/2015-05-22/1612/277336",
  "pool": null,
  "resolved_macros": null,
  "name": null,
  "created_at": "2015-05-22T10:28:20Z",
  "command": {
    "action": "restore",
    "private_dns": null,
  }
}
```

```
"cluster_inst_id": 23528,
"parameters": {
    "table_names": "tsdb",
    "backup_id": "backup_1432284975205",
    "automatic": true,
    "overwrite": true,
    "s3://hbasetabledata.com/opentsdb_backup_22_5_2015"
}
},
"end_time": null,
"timeout": null
}
```

4.7.19 Update an HBase Snapshot Schedule

PUT /api/v1.3/clusters/(string: id_or_label) /snapshot_schedule

This API is used to change attributes of a schedule that takes snapshots in an HBase cluster.

Note: A cluster must be running to update an HBase snapshot schedule.

Parameters

Note: Parameters marked in **bold** are mandatory. Others are optional and have default values.

Parameter	Description
frequency_unit	Frequency unit for the HBase snapshot schedule. Input is a hash {"unit":"value"}. Accepted unit is minutes, hours, days, weeks, and months.
frequency_num	The number of concurrent snapshots configured to be created in a snapshot schedule. The default value is 1.
s3 location	The S3 location that contains the snapshot.

Example

Request

```
curl -X PUT -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" -H "Content-Type:application/json" -H "Accept: application/json" -d '{snapshot-schedule {
    "frequency_unit" => "hours",
    "frequency_num" => 20,
    "s3_location" => "s3://location//", }
}'
```

Response

```
{"status":"SUSPENDED","frequency_num":1200,"frequency_unit":"mins","id":206,"s3_location":"s3://location//",}
```

4.7.20 Launch a new cluster (deprecated)

Note: This API has been deprecated. Instead, use the [Start or Terminate a Cluster](#).

Example

Goal

To launch a new cluster.

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" https://api.qubole.com/api/${V}/hadoop_cluster"
```

Response

```
{"message": "Launching cluster. It will take 1-2 minutes."}
```

Note: Trying to relaunch a cluster if it already exists will not have any effect.

4.7.21 Check Cluster Status (deprecated)

GET /api/v1.3/hadoop_cluster

Note: This API has been deprecated. Instead, use the [Check Cluster Status](#).

Parameters

There are no parameters, that are required, for this API.

Response

The response contains a JSON object representing the cluster. All the attributes mentioned here will be returned (except when otherwise specified or redundant).

Example

Goal

Check the status of the cluster.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json" https://api.qubole.com/api/v1.3/hadoop_cluster"
```

Response

The response can be one of the following:

- If the cluster is *down*

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
  "cluster": {
    "state": "DOWN"
  }
}
```

- If the cluster is *being brought up*

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
{
  "uptime": null,
  "tasktracker": {
    "spot": 0,
    "running": 0,
    "pending": 0
  },
  "state": "PENDING",
  "jt_web": null,
  "launch_ts": 1403256709,
  "nodes": []
}
```

- If the cluster is *up and running*

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
    "uptime": "less than a minute",
    "master_public_dns": "ec2-xx-yy-zz-qq.compute-1.amazonaws.com",
    "tasktracker": {
        "spot": 0,
        "running": 1,
        "pending": 0
    },
    "state": "UP",
    "jt_web": "https://api.qubole.com/qpal/handle_proxy?query=http%3A%2F%2Fec2-xx-yy-zz-qq.compute-1.amazonaws.com",
    "launch_ts": 1403256709,
    "nodes": [
        {
            "private_ip": "domU-xx-yy-zz-qq-aa-dd.compute-1.internal",
            "instance_id": "i-abcdefg",
            "alias": "master",
            "instance_type": "m1.large",
            "is_spot": false,
            "launch_ts": 1403256755,
            "public_dns": "ec2-xx-yy-zz-qq.compute-1.amazonaws.com"
        },
        {
            "private_ip": "ip-aa-bb-cc-dd.ec2.internal",
            "instance_id": "i-pqrstuv",
            "alias": "node0001",
            "instance_type": "m1.xlarge",
            "is_spot": false,
            "launch_ts": 1403256755,
            "public_dns": "ec2-pp-qq-rr-zz.compute-1.amazonaws.com"
        }
    ]
}
```

```

        }
    ]
}

```

4.7.22 Terminate a cluster (deprecated)

PUT /api/v1.2/clusters/

Note: This API has been deprecated. Use the *Start or Terminate a Cluster* instead.

Parameters

Parameter	Description
Status	Kill

Example

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Content-Type: application/json" -H "Accept: application/json"
```

Response

```
`` {"message": "Terminating cluster. It will take 1-2 minutes."}``
```

Note: Issuing the kill action when a cluster is not running, will not have any effect.

4.8 DbTap API

4.8.1 Create a DbTap

Resource URI	db_taps/
Request Type	POST
Supporting Versions	v1.2
Return Value	Json object representing the newly created DbTap.

Parameter	Description
db_name	Database Name. This is the data store name that will be created in the QDS.
db_host	IP address or hostname of the data store.
db_user	User name to login to the data store.
db_passwd	Password to login to the data store.
db_port	TCP Port to connect on. If not specified, then the default port for the datastore type will be used.
db_type	Type of database. Valid values are mysql, vertica, mongo, postgresql and redshift. Default is mysql.
db_location	Location of database. Valid values are us-east-1, us-west-2, ap-southeast-1, eu-west-1, on-premise.
gateway_ip	IP address or hostname of the gateway machine.
gateway_username	User name to login to the gateway machine.
gate-way_private_key	Private key for the aforementioned user to login to the gateway machine.

Note

- Gateway parameters (*gateway_ip*, *gateway_username*, *gateway_private_key*) can be specified only if *db_location* is ‘on-premise’.
- If you do not want to use a gateway machine to access your data store, you need not specify any of the gateway parameters.
- Though the gateway parameters are optional, if any one of the gateway parameters is specified then all three must be specified.
- Port 22 must be open on the gateway machine and it must have access to the data store.

Example

Payload

```
{  
  "db_name": "doc_example",  
  "db_host": "localhost",  
  "db_user": "doc_writer",  
  "db_passwd": "",  
  "db_type": "mysql",  
  "db_location": "us-east-1"  
}
```

```
curl -i -X POST -H "Content-Type: application/json" -H "Accept: application/json" -H "X-AUTH-TOKEN: S...
```

Sample Response

```
{  
  "account_id": 00000,  
  "active": true,  
  "db_user": "doc_writer",  
  "user_id": 1,  
  "db_passwd": "",  
  "db_name": "doc_example",  
  "created_at": "2013-03-15T10:02:42Z",  
  "db_host": "localhost",  
  "db_location": "us-east-1",  
  "db_type": "mysql"  
  "id": 3,  
  "port": null  
}
```

Take a note of the DbTap id (in this case 3). It will be used in later examples.

```
export DBTAPID=3
```

4.8.2 Delete a DbTap

DELETE /api/v1.2/db_taps/ (int: dbtap_id) /

Deletes a *Data Store*.

Response

The response contains a JSON object with the status of the operation.

Example

Goal: delete a *Data Store* having id 123

```
curl -i -X DELETE -H "Content-Type: application/json" \
-H "Accept: application/json" -H "X-AUTH-TOKEN:$X_AUTH_TOKEN" \
https://api.qubole.com/api/v1.2/db_taps/123
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{ "status": "deleted" }
```

4.8.3 Edit a DbTap

Resource URI	db_taps/ <i>id</i>
Request Type	PUT
Supporting Versions	v1.2
Return Value	Json object representing the updated DbTap.

Parameter	Description
db_name	Database Name. This is the data store name that is created in the QDS.
db_host	IP address or hostname of the data store.
db_user	User name to login to the data store.
db_passwd	Password to login to the data store.
db_port	TCP Port to connect on.
db_type	Type of database. Valid values are mysql, postgresql, mongo, redshift and vertica.
db_location	location of database. Valid values are us-east-1, us-west-2, eu-west-1, ap-singapore-1, on-premise.

Example

```
curl -i -X PUT -H "Content-Type: application/json" -H "Accept: application/json" -H "X-AUTH-TOKEN: $X_AUTH_TOKEN" -d { "status": "deleted" } https://api.qubole.com/api/v1.2/db_taps/123
```

Sample Response

```
{
  "account_id": 00000,
  "active": true,
  "db_user": "doc_writer",
  "user_id": 1,
  "db_passwd": "",
  "db_name": "doc_example",
  "created_at": "2013-03-15T10:02:42Z",
  "db_type": "mysql",
  "db_location": "us-east-1"
```

```
"db_host": "localhost",
"id": 3,
"port": 3306
}
```

4.8.4 List DbTaps

Resource URI	db_taps
Request Type	GET
Supporting Versions	v1.2
Return Value	JSON array of DbTaps.

Example

```
curl -i -X GET -H "Content-Type: application/json" -H "Accept: application/json" -H "X-AUTH-TOKEN: $A
```

Sample Response

```
{
  "paging_info": {
    "next_page": null,
    "per_page": 10,
    "previous_page": null
  },
  "db_taps": [
    {
      "active": true,
      "account_id": 00000,
      "user_id": 1,
      "db_user": "root",
      "db_passwd": "",
      "db_location": "us-east-1",
      "db_type": "mysql",
      "db_name": "jenkins",
      "created_at": "2012-12-27T17:20:32Z",
      "db_host": "localhost",
      "port": null,
      "id": 1
    },
    {
      "active": true,
      "account_id": 00000,
      "user_id": 1,
      "db_user": "doc_writer",
      "db_passwd": "",
      "db_name": "doc_example",
      "created_at": "2013-03-15T10:02:42Z",
      "db_location": "us-east-1",
      "db_type": "postgresql",
      "db_host": "localhost",
      "port": null,
      "id": 3
    }
  ]
}
```

```
[  
}
```

4.8.5 List tables in a DbTap

Resource URI	db_taps/ <i>id</i> /tables
Request Type	GET
Supporting Versions	v1.2
Return Value	JSON Array of tables in a DbTap.

Example

```
curl -i -X GET -H "Content-Type: application/json" -H "Accept: application/json" -H "X-AUTH-TOKEN: $A
```

Sample Response

```
[ "CUSTOMER", "LINEITEM", "NATION", "ORDERS", "PART", "PARTSUPP", "REGION", "SUPPLIER" ]
```

4.8.6 View a DbTap

Resource URI	db_taps/ <i>id</i>
Request Type	GET
Supporting Versions	v1.2
Return Value	Json object representing the DbTap.

Example

```
curl -i -X GET -H "Content-Type: application/json" -H "Accept: application/json" -H "X-AUTH-TOKEN: $A
```

Sample Response

```
{
  "account_id": 00000,
  "active": true,
  "db_user": "doc_writer",
  "user_id": 1,
  "db_location": "us-east-1",
  "db_type": "mysql",
  "db_passwd": "",
  "db_name": "doc_example",
  "created_at": "2013-03-15T10:02:42Z",
  "db_host": "localhost",
  "id": 3,
  "port": null
}
```

4.9 Scheduler API

4.9.1 Create a Schedule

POST /api/v1.2/scheduler/

This API creates a new schedule to run commands automatically at certain frequency in a specified interval.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
command_type	A valid command type supported by Qubole. For example, HiveCommand, HadoopCommand, PigCommand.
command	JSON object describing the command. Refer to the Command API for more details. Sub fields can use macros. Refer to the Qubole Scheduler for more details.
start_time	Start datetime for the schedule
end_time	End datetime for the schedule
frequency	Specify how often the job should run. Input is an integer. Accepted unit are minutes, hours, days, weeks, or months. For example, frequency of one hour is represented as { "hours": "1" }
name	A user-defined name for a schedule. If name is not specified, then a system-generated Schedule ID is set as the name.
macros	Expressions to evaluate macros. Macros can be used in parameterized commands. Refer to the Macros in Scheduler page for more details.
no_catch_up	Set this parameter to <code>true</code> if you want to skip instances supposed to have run in the past and run only the latest instances. By default, this parameter is set to <code>false</code> . When a new schedule is created, the scheduler runs instances from start time to the current time. For example, if a daily schedule is created from Jun 1, 2015 on Dec 1, 2015, jobs are run for Jun 1, 2015, Jun 2, 2015, and so on. If you do not want the scheduler to run the missed instances for months earlier to Dec, set <code>no_catch_up</code> to <code>true</code> .
time_zone	Timezone of the start and end time of the schedule. Scheduler will understand ZoneInfo identifiers. For example, Asia/Kolkata. For a list of identifiers, check column 3 in List of TZ in databases . Default value is UTC.
time_out	Unit is minutes. A number that represents a maximum amount of time the job should wait for dependencies to be satisfied.
concurrency	Specify how many job actions can run at a time. Default value is 1.
dependency_info	Describe dependencies for this job. Check the Hive Datasets as Schedule Dependency for more information.
<i>Notification Parameters</i>	It is an optional parameter that is set to false by default. You can set it to true if you want to be notified through email about instance failure. Notification Parameters provides more information.

Notification Parameters

Parameter	Description
is_digest	It is a notification email type that is set to true if a job periodicity is in minutes or hours. If it set to false, the email type is immediate by default.
notify_failure	If this option is set to true, you receive job failure notifications.
notify_success	If this option is set to true, you receive job success notifications.
notification_email_list	By default, the current user's email ID is added. You can add additional email IDs as required.

Response

The response contains a JSON object representing the created schedule.

Example

Goal: Create a new schedule to run Hive queries.

Use the following query as shown in the example below:

```
CREATE EXTERNAL TABLE daily_tick_data (
    date2 string,
    open float,
    close float,
    high float,
    low float,
    volume INT,
    average FLOAT)
PARTITIONED BY (
    stock_exchange STRING,
    stock_symbol STRING,
    year STRING,
    date1 STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION 's3n://paid-qubole/default-datasets/stock_ticker';
```

date1 is the date in the format **YYYY-MM-DD**

The dataset is available from **2012-07-01**.

For this example, let us assume that the dataset is updated everyday at 1AM UTC, and the jobs are scheduled at 2AM UTC, everyday.

The query shown below aggregates the data for every stock symbol, every day.

```
payload:
{
  "command_type": "HiveCommand",
  "command": {
    "query": "select stock_symbol, max(high), min(low), sum(volume) from daily_tick_data where date1 = "
  },
  "macros": [
    {
      "formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"
    }
  ],
}
```

```

"start_time": "2012-07-01T02:00Z",
"end_time": "2022-07-01T02:00Z",
"frequency": 1,
"time_unit": "days",
"time_out": "10",
"dependency_info": {}
}

```

Command

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json"
-H "Content-type: application/json" --data @payload
"https://api.qubole.com/api/v1.2/scheduler"
```

Sample Response

```
{
  "time_out": 10,
  "status": "RUNNING",
  "start_time": "2012-07-01 02:00",
  "label": "default",
  "concurrency": 1,
  "frequency": 1,
  "no_catch_up": false,
  "template": "generic",
  "command": {
    "sample": false, "loader_table_name": null, "md_cmd": null, "script_location": null, "approx_mode": null
  },
  "time_zone": "UTC",
  "time_unit": "days",
  "end_time": "2022-07-01 02:00",
  "user_id": 108,
  "macros": [{"formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"}],
  "incremental": {},
  "command_type": "HiveCommand",
  "name": "3159",
  "dependency_info": {},
  "id": 3159,
  "next_materialized_time": null
}
```

Take a note of the schedule id (in this case 3159). It will be used in other examples.

```
export SCHEDID=3159
```

4.9.2 View a Schedule

GET /api/v1.2/scheduler/ (int: id)

This API is used to view an existing schedule that is created to run commands automatically at certain frequency in a specified interval.

Resource URI	scheduler/ <i>id</i>
Request Type	GET
Supporting Versions	v1.2
Return Value	Json object representing the schedule.

Example

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" -H "Content-type: application/json"
```

Response

```
{
  "concurrency": 1,
  "time_unit": "days",
  "command": {
    "approx_mode": false,
    "query": "select stock_symbol, max(high), min(low), sum(volume) from daily_tick_data where date1=2012-07-01 and date2=2012-07-02 group by stock_symbol",
    "approx_aggregations": false,
    "sample": false
  },
  "user_id": 39,
  "dependency_info": {
    "hive_tables": [
      {
        "window_end": "0",
        "initial_instance": "2012-07-01T00:00Z",
        "name": "daily_tick_data",
        "interval": {
          "days": "1"
        },
        "columns": {
          "stock_symbol": [
            "ibm",
            "orcl"
          ],
          "stock_exchange": [
            "nasdaq",
            "nyse"
          ]
        },
        "window_start": "-1",
        "time_zone": "UTC"
      }
    ]
  },
  "time_out": 10,
  "macros": [
    {
      "formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"
    }
  ],
  "end_time": "2022-07-01 02:00",
  "start_time": "2012-07-01 02:00",
  "frequency": 1,
  "id": 2266,
  "time_zone": "UTC",
  "command_type": "HiveCommand",
  "status": "RUNNING"
}
```

4.9.3 List Schedules

GET /api/v1.2/scheduler/

This API is used to list all existing schedules created to run commands automatically at certain frequency in a specified interval.

Resource URI	scheduler/
Request Type	GET
Supporting Versions	v1.2
Return Value	Json array of schedules. All schedules in all states are shown.

Example

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" -H "Content-type: application/json"
```

Sample Response

```
{
  "paging_info": {
    "previous_page": null,
    "per_page": 10,
    "next_page": null
  },
  "schedules": [
    {
      "concurrency": 1,
      "time_unit": "days",
      "command": {
        "approx_mode": false,
        "query": "select stock_symbol, max(high), min(low), sum(volume) from daily_tick_data where date >= '2012-07-01' and date <= '2012-07-02' group by stock_symbol",
        "approx_aggregations": false,
        "sample": false
      },
      "user_id": 39,
      "dependency_info": {
        "hive_tables": [
          {
            "initial_instance": "2012-07-01T00:00Z",
            "window_end": "0",
            "name": "daily_tick_data",
            "interval": {
              "days": "1"
            },
            "time_zone": "UTC",
            "window_start": "-1",
            "columns": {
              "stock_symbol": [
                "ibm",
                "orcl"
              ],
              "stock_exchange": [
                "nasdaq",
                "nyse"
              ]
            }
          }
        ]
      }
    }
  ]
}
```

```
        }
    ],
},
"time_out": 10,
"macros": [
{
    "formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"
}
],
"end_time": "2022-07-01 02:00",
"start_time": "2012-07-01 02:00",
"frequency": 1,
"id": 2266,
"time_zone": "UTC",
"command_type": "HiveCommand",
"status": "PREP"
}
{
"dependency_info": {
    "hive_tables": [
    {
        "initial_instance": "2012-5-12T00:00Z",
        "window_start": -1,
        "columns": {
            "country": [
                "IND",
                "US",
                "CAN"
            ]
        },
        "interval": {
            "minutes": 60
        },
        "time_zone": "UTC",
        "window_end": 0,
        "name": "demo_data3"
    }
]
},
"status": "KILLED",
"user_id": 39,
"time_zone": "UTC",
"time_out": 10,
"macros": "[{"formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"}]",
"command": {
    "account_id": 14,
    "created_at": "2013-03-06T04:51:37Z",
    "id": 863,
    "halt_on_failure": false,
    "updated_at": "2013-03-06T04:51:37Z"
},
"command_type": "CompositeCommand",
"start_time": "2012-05-12T00:00:00Z",
"end_time": "2015-01-01T00:00:00Z",
"frequency": {"days":1},
"id": 1929,
"concurrency": 1
},
```

```
{
  "dependency_info": {
    "hive_tables": null
  },
  "status": "RUNNING",
  "user_id": 39,
  "time_zone": "UTC",
  "time_out": 10,
  "macros": "[{"formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"}]",
  "command": {
    "account_id": 14,
    "created_at": "2013-03-05T11:13:00Z",
    "id": 647,
    "halt_on_failure": false,
    "updated_at": "2013-03-05T11:13:00Z"
  },
  "command_type": "CompositeCommand",
  "start_time": "2012-05-12T00:00:00Z",
  "end_time": "2015-01-01T00:00:00Z",
  "frequency": {"days": 1},
  "id": 1922,
  "concurrency": 1
}
]
```

4.9.4 Edit a Schedule

PUT /api/v1.2/scheduler/(Scheduler ID)

This API is used to edit an existing schedule that is created to run commands automatically in a specified interval. You can edit a schedule by sending a PUT request with attributes that you want to modify.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
command_type	A valid command type supported by Qubole. For example, HiveCommand, HadoopCommand, PigCommand.
command	JSON object describing the command. Refer to the Command API for more details. Sub fields can use macros. Refer to the Qubole Scheduler for more details.
name	A user-defined name for a schedule. If name is not specified, then a system-generated Schedule ID is set as the name.
start_time	Start datetime for the schedule
end_time	End datetime for the schedule
frequency	Specify how often the job should run. Input is an integer. Accepted units are minutes, hours, days, weeks, or months. For example, frequency of one hour is represented as <code>{"hours": "1"}</code>
macros	Expressions to evaluate macros. Macros can be used in parameterized commands. Refer to the Macros in Scheduler page for more details.
no_catch_up	Set this parameter to <code>true</code> if you want to skip instances supposed to have run in the past and run only the latest instances. By default, this parameter is set to <code>false</code> . When a new schedule is created, the scheduler runs instances from start time to the current time. For example, if a daily schedule is created from Jun 1, 2015 on Dec 1, 2015, jobs are run for Jun 1, 2015, Jun 2, 2015, and so on. If you do not want the scheduler to run the missed instances for months earlier to Dec, set <code>no_catch_up</code> to <code>true</code> .
time_zone	Timezone of the start and end time of the schedule. Scheduler will understand ZoneInfo identifiers. For example, Asia/Kolkata. For a list of identifiers, check column 3 in List of TZ in databases . Default value is UTC.
time_out	Unit is minutes. A number that represents a maximum amount of time the job should wait for dependencies to be satisfied.
concurrency	Specify how many job actions can run at a time. Default value is 1.
dependency_info	Describe dependencies for this job. Check the Hive Datasets as Schedule Dependency for more information.
<i>Notification Parameters</i>	It is an optional parameter that is set to false by default. You can set it to true if you want to be notified through email about instance failure. Notification Parameters provides more information.

Notification Parameters

Parameter	Description
is_digest	It is a notification email type that is set to true if a job periodicity is in minutes or hours. If it set to false, the email type is immediate by default.
notify_failure	If this option is set to true, you receive job failure notifications.
notify_success	If this option is set to true, you receive job success notifications.
notification_email_list	By default, the current user's email ID is added. You can add additional email IDs as required.

Response

The response contains a JSON object representing the edited schedule.

Example

Goal: Modify a schedule to run every 2 hours

```
Payload:
{
    frequency: 30
    time_unit: days
}
```

Command

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json"
-H "Content-type: application/json" --data @payload
"https://api.qubole.com/api/v1.2/scheduler/3159"
```

Response

```
{
  "email_list": "qubole@qubole.com",
  "dependency_info": {},
  "end_time": "2022-07-01 02:00",
  "status": "RUNNING",
  "no_catch_up": false,
  "label": "default",
  "concurrency": 1,
  "frequency": 30,
  "time_zone": "UTC",
  "template": "generic",
  "command": {
    "sample": false,
    "loader_table_name": null,
    "md_cmd": null,
    "approx_mode": false,
    "query": "select * from table",
    "user_id": 108,
    "is_digest": false,
    "time_unit": "days",
    "digest_time_hour": 0,
    "macros": [{"formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"}],
    "incremental": {},
    "bitmap": 0,
    "digest_time_minute": 0,
    "can_notify": false,
    "command_type": "HiveCommand",
    "script": null
  }
}
```

```
"name": "3159",
"start_time": "2012-07-01 02:00",
"time_out": 10,
"id": 3159,
"next_materialized_time": "2012-07-07 02:00"
}
```

4.9.5 Suspend,Resume or Kill a Schedule

PUT /api/v1.2/scheduler/ (int: id)

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

This API is used to suspend, resume, or kill an existing schedule created to run commands automatically at certain frequency in a specified interval.

Resource URI	scheduler/ <i>id</i>
Request Type	PUT
Supporting Versions	v1.2
Return Value	JSON object with the status of the operation.

Parameters

Parameter	Description
status	Valid values or <i>suspend,resume</i> or <i>kill</i> .

Examples

Suspend

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" -H "Content-type: application/json" -d '{"status": "SUSPENDED"}'
```

Response

```
{"succeeded": "true", "status": "SUSPENDED"}
```

Resume

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" -H "Content-type: application/json" -d {"status": "RESUMED"}
```

Response

```
{"succeeded": "true", "status": "RUNNING"}
```

Kill

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" -H "Content-type: application/json" -d '{"action": "KILL"}' https://qubole.com/api/v1.2/scheduler/{SchedulerID}/kill
```

Response

```
{"succeeded": "true", "status": "KILLED"}
```

4.9.6 Clone a Schedule

POST /api/v1.2/scheduler/ (SchedulerID) /duplicate

This API is used to clone an existing schedule by providing a new schedule name.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
command_type	A valid command type supported by Qubole. For example, HiveCommand, HadoopCommand, PigCommand.
command	JSON object describing the command. Refer to the Command API for more details. Sub fields can use macros. Refer to the Qubole Scheduler for more details.
start_time	Start datetime for the schedule
end_time	End datetime for the schedule
frequency	Specify how often the job should run. Input is an integer. Accepted unit are minutes, hours, days, weeks, or months. For example, frequency of one hour is represented as { "hours": "1" }
name	A user-defined name for a schedule. If name is not specified, then a system-generated Schedule ID is set as the name. While cloning an existing schedule, you must change the name.
macros	Expressions to evaluate macros. Macros can be used in parameterized commands. Refer to the Macros in Scheduler page for more details.
no_catch_up	Set this parameter to <code>true</code> if you want to skip instances supposed to have run in the past and run only the latest instances. By default, this parameter is set to <code>false</code> . When a new schedule is created, the scheduler runs instances from start time to the current time. For example, if a daily schedule is created from Jun 1, 2015 on Dec 1, 2015, jobs are run for Jun 1, 2015, Jun 2, 2015, and so on. If you do not want the scheduler to run the missed instances for months earlier to Dec, set <code>no_catch_up</code> to <code>true</code> .
time_zone	Timezone of the start and end time of the schedule. Scheduler will understand ZoneInfo identifiers. For example, Asia/Kolkata. For a list of identifiers, check column 3 in List of TZ in databases . Default value is UTC.
time_out	Unit is minutes. A number that represents a maximum amount of time the job should wait for dependencies to be satisfied.
concurrency	Specify how many job actions can run at a time. Default value is 1.
dependency_info	Describe dependencies for this job. Check the Hive Datasets as Schedule Dependency for more information.
<i>Notification Parameters</i>	It is an optional parameter that is set to <code>false</code> by default. You can set it to <code>true</code> if you want to be notified through email about instance failure. Notification Parameters provides more information.

Notification Parameters

Parameter	Description
is_digest	It is a notification email type that is set to true if a job periodicity is in minutes or hours. If it set to false, the email type is immediate by default.
notify_failure	If this option is set to true, you receive job failure notifications.
notify_success	If this option is set to true, you receive job success notifications.
notification_email_list	By default, the current user's email ID is added. You can add additional email IDs as required.

Response

The response contains a JSON object representing the cloned schedule.

Example

Goal: Clone an existing schedule, for example: schedule ID 3159, to create a new schedule. For more information on how to create a schedule, see [Create a Schedule](#).

While creating a schedule, we created a schedule that aggregates data every day, for every stock symbol, and for each stock exchange. For example, if you want to edit the query to also calculate the total transaction amount for the stock in a day, provide the following query.

```
{
  "command_type": "HiveCommand",
  "command": {
    "query": "select stock_symbol, stock_exchange, max(high), min(low), sum(volume) from daily_tick_data
  },
  "start_time": "2012-11-01T02:00Z",
  "end_time": "2022-10-01T02:00Z"
}
```

Command

```
Payload:
{
  name: schedule1
}
```

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json"
-H "Content-type: application/json" --data @payload
"https://api.qubole.com/api/v1.2/scheduler/3159/duplicate"
```

Sample Response

```
{
  "time_out": 10,
  "status": "RUNNING",
  "start_time": "2012-07-01 02:00",
  "label": "default",
  "concurrency": 1,
  "frequency": 1,
  "no_catch_up": false,
  "template": "generic",
  "command": {
    "sample": false,
    "loader_table_name": null,
    "md_cmd": null,
    "script_location": null,
    "approx_mode": null
  }
}
```

```
        },
        "time_zone": "UTC",
        "time_unit": "days",
        "end_time": "2022-07-01 02:00",
        "user_id": 108,
        "macros": [{"formatted_date": "Qubole_nominal_time.format('YYYY-MM-DD')"}],
        "incremental": {},
        "command_type": "HiveCommand",
        "name": "schedule1",
        "dependency_info": {},
        "id": 3160,
        "next_materialized_time": null
    }
```

4.9.7 List Schedule Actions

GET /api/v1.2/scheduler/ (int: id) /actions

Retrieves a list of actions run for a scheduler. The list is paginated.

Response

The list contains information about commands that are run as part of the action. The list is ordered with latest sequence first.

Example

Goal: Retrieve a list of actions for a schedule. The list has 3 actions per page.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/scheduler/${SCHEID}/actions?per_page=3"
```

Response

```
{
    "actions": [
        {
            "status": "done",
            "is_rerun_of": 47519,
            "nominal_time": "2014-06-26T11:00:00Z",
            "done": true,
            "sequence_id": 4226,
            "query_hist_id": 277791,
            "rerun_number": 2,
            "id": 47520,
            "dependencies": {
                "not_found": [
                    ],
                "found": [
                    ]
            }
        }
    ]
}
```

```

        ],
    },
    .
    .
    .
    "periodic_job_id": 30562,
},
{
    "done": true,
    "sequence_id": 4226,
    "query_hist_id": 277790,
    .
    .
    .
    "status": "done",
},
{
    "status": "done",
    "is_rerun_of": null,
    "periodic_job_id": 30562,
}
],
"paging_info": {
    "previous_page": null,
    "per_page": "3",
    "next_page": 2
}
}
}

```

4.9.8 View a Schedule's Action

GET /api/v1.2/scheduler/ (int: id) /actions/
int: sequence_id

This API is used to view an existing schedule's action.

sequence_id is the nth action of the schedule. If an action has been rerun, then both the actions are returned.

Response

Response is a list of scheduler actions. Each action may or may not have an empty command child object. A command may be empty if dependencies were not satisfied.

Example

For brevity, the JSON is truncated.

Goal: View details of the action of a schedule.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/scheduler/${SCHEDID}/actions/1/"
```

Response

```
{  
    "actions": [  
        {  
            "sequence_id": 1,  
            "status": "done",  
            "is_rerun_of": null,  
            "periodic_job_id": 30562,  
            "created_at": "2014-05-26T11:31:22Z",  
            "nominal_time": "2014-01-01T10:00:00Z",  
            "done": true,  
            "query_hist_id": 241003,  
            "rerun_number": 1,  
            "id": 37129,  
            "dependencies": {  
                },  
            "command": {  
                "status": "done",  
                "command": {  
                    "loader_stable": null,  
                    .  
                    .  
                    .  
                    "approx_mode": false  
                },  
                .  
                .  
                .  
            },  
        },  
        ],  
        "paging_info": {  
            "previous_page": null,  
            "per_page": 10,  
            "next_page": null  
        }  
    }  
}
```

Goal: Retrieve an action which has been rerun. In this example, the action ID 4226 is retrieved.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \  
-H "Accept: application/json" \  
-H "Content-type: application/json" \  
"https://api.qubole.com/api/v1.2/scheduler/${SCHEDID}/actions/4226/"
```

Response

```
{  
    "actions": [  
        {  
            "sequence_id": 4226,  
            "status": "done",  
            .  
            .  
            .  
            "is_rerun_of": 47519,
```

```

        }
    },
    {
        "status": "done",
        "is_rerun_of": null,
        .
        .
        .
        "sequence_id": 4226,
    }
],
"paging_info": {
    "previous_page": null,
    "per_page": 10,
    "next_page": null
}
}

```

4.9.9 Kill a Schedule Action

PUT /api/v1.2/actions/ (int: id) /kill

Cancels a running scheduled action. The *ID* can be obtained by listing the actions using any one of the [List Schedule Actions](#) or the [List All Actions](#).

Response

A JSON hash that encodes if the operation is a success or not.

```
{"kill_succeeded":"true"}
```

Example

Goal: Rerun an action with id \${ACTIONID}

```
curl -i -X PUT -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/actions/${ACTIONID}/kill"
```

Response

```
{"kill_succeeded":"true"}
```

4.9.10 Rerun a Scheduled Action

PUT /api/v1.2/actions/ (int: id) /rerun

Reruns a scheduled action. The *ID* can be obtained by listing the actions using any one of the [List Schedule Actions](#) or the [List All Actions](#).

Response

A JSON hash that encodes if the operation is successful or not.

```
{"status": "rescheduled"}
```

Example

Goal: Rerun an action with id \${ACTIONID}

```
curl -i -X POST -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/actions/${ACTIONID}/rerun"
```

Response

```
{"status": "rescheduled"}
```

4.9.11 List All Actions

GET /api/v1.2/actions

Retrieves a list of actions run by the scheduler. The list can belong to any schedule in the account. It is ordered in reverse chronological order. The list is also paginated.

Response

The list contains information about commands that are run as part of the action.

Example

Goal: Retrieve a list of actions for a schedule. The list has 3 actions per page.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/actions?per_page=3"
```

Response

```
{
  "actions": [
    {
      "status": "done",
      "is_rerun_of": 47519,
      "nominal_time": "2014-06-26T11:00:00Z",
      "done": true,
      "sequence_id": 4226,
      "query_hist_id": 277791,
      "rerun_number": 2,
```

```

    "id": 47520,
    "dependencies": {
        "not_found": [
            ],
        "found": [
            ]
    },
    .
    .
    .
    "periodic_job_id": 30562,
},
{
    "done": true,
    "sequence_id": 4226,
    "query_hist_id": 277790,
    .
    .
    .
    "status": "done",
},
{
    "status": "done",
    "is_rerun_of": null,
    "periodic_job_id": 30562,
}
],
"paging_info": {
    "previous_page": null,
    "per_page": "3",
    "next_page": 2
}
}

```

4.9.12 View an Action

GET /api/v1.2/actions/ (int: id)

Response

Response is a scheduled action. The action may or may not have an empty command child object. A command may be empty if dependencies were not satisfied.

This API differs from the [View a Schedule's Action](#). In that API, instead of the *sequence_id* of the action, it accepts the unique ID of the action within the schedule.

Example

Goal: View the details about the first action of a schedule.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
```

```
-H "Content-type: application/json" \
"https://api.qubole.com/api/v1.2/actions/37129/"
```

Response

```
{
  "status": "done",
  "done": true,
  "command": {
    "meta_data": {
      "logs_resource": "commands/165025/logs",
      "results_resource": "commands/165025/results"
    },
    "status": "done",
    "command_source": "SCHEDULED",
    "progress": 100,
    "qbol_session_id": 49007,
    "command": {
      "approx_mode": false,
      "md_cmd": false,
      "loader_stable": null,
      "script_location": null,
      "query": "select stock_symbol, max(high), min(low), sum(volume) from daily_tick_data where date > '2012-07-01' and date < '2012-07-02' group by stock_symbol",
      "sample": false,
      "loader_table_name": null,
      "approx_aggregations": false
    },
    "created_at": "2015-04-15T10:05:11Z",
    "start_time": 1429092315,
    "end_time": 1429092346,
    "command_type": "HiveCommand",
    "pid": 12110,
    "account_id": 3,
    "label": "default",
    "template": "generic",
    "timeout": null,
    "can_notify": false,
    "pool": null,
    "user_id": 3,
    "submit_time": 1429092311,
    "name": "",
    "id": 165025,
    "path": "/tmp/2015-04-15/3/165025",
    "qlog": "{\"QBOL-QUERY-SCHEMA\":{\"/tmp/2015-04-15/3/165025.dir/000\":[{\"ColumnType\":\"string\\n\"}],\"num_result_dir\":1,\"resolved_macros\":{\"Qubole_nominal_time\":\"Sun Jul 01 2012 02:00:00 GMT+0000\\\",\"formatted_date\":true}},\"qboltypes\":[]}",
    "dependencies": {
      "not_found": [],
      "found": []
    },
    "nominal_time": "2012-07-01T02:00:00Z",
    "is_rerun_of": null,
    "sequence_id": 1,
    "query_hist_id": 165025,
    "job_id": 3094
  }
}
```

```
"rerun_number": 1,
"id": 46096
}
```

4.10 Reports API

4.10.1 All Commands Report

GET /api/v1.2/reports/all_commands

This API retrieves the **All Commands** report containing the query metrics in JSON format.

Note: The following points are related to a report API:

- If the difference between start date and end date is more than 60 days, then the system defaults to 1 month window from the current day's date.
- If either start date or end date is not provided, then the system defaults to 1 month window from the current day's date.
- If you want to get data for a window more than 2 months, then write an email to solutions@quoble.com.

Parameters

Parameter	Description
start_date	The date from which you want the report (inclusive). The API default is 30 days before the end date. This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
end_date	The date until which you want the report (inclusive). The API default is <i>today</i> . This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
offset	The starting point of the results. The API default is <i>0</i> .
limit	The number of results to fetch. The API default is <i>10</i> .
sort_column	The column used to sort the report. The Valid choices are <i>time</i> , <i>cpu</i> , <i>fs_bytes_read</i> , <i>fs_bytes_written</i> . The API default is, <i>time</i> (chronological order).
by_user	Report only those queries which are created by the current user. By default all queries by the current account are reported.

Response Parameters

Parameter	Description
start_date	The starting date of the report.
end_date	The ending date of the report.
sort_column	The sort column used.

An array of:

id	The query ID of the command.
created_at	The time when the query was created.
submitted_by	The Email address of the user who created the query.
command_type	The type of the command (HiveCommand, PrestoCommand, and so on.)
command_summary	The summary of the command (query/latin_statements/script_location, and so on.)
status	The status of the command (whether it succeeded or failed, and so on.)
cpu	The total cumulative CPU, (in ms), consumed by this command.
fs_bytes_read	The total bytes read by this command.
fs_bytes_written	The total bytes written by this command.

Examples

Goal

To get the default report.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/all_commands"
```

Goal

To get the report for commands executed only by the current user.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/all_commands?by_user"
```

Goal

To get the report for commands executed during a specific time period and sorted by total bytes read.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/all_commands?start_date=2014-04-01&end_date=2014-04-21&sort_
```

Sample Response

```
{
  "end_date": "2014-04-21T10:00:00Z",
  "sort_column": "fs_bytes_read",
  "queries": [
    {
      "id": 198956,
      "cpu": 14830,
      "created_at": "2014-04-01T07:01:01Z",
      "fs_bytes_written": 880829,
      "command_type": "HadoopCommand",
      "submitted_by": "demo1@qubole.com",
      "fs_bytes_read": 3593584,
      "status": "done",
      "command_summary": "'s3://paid-qubole/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar'"
    },
    {
      "id": 198957,
      "cpu": 14830,
      "created_at": "2014-04-01T07:01:01Z",
      "fs_bytes_written": 880829,
      "command_type": "HadoopCommand",
      "submitted_by": "demo1@qubole.com",
      "fs_bytes_read": 3593584,
      "status": "done",
      "command_summary": "'s3://paid-qubole/HadoopAPIExamples/jars/hadoop-0.20.1-dev-streaming.jar'"
    }
  ]
}
```

```

    "id": 198954,
    "cpu": 6560,
    "created_at": "2014-04-01T06:58:42Z",
    "fs_bytes_written": 18,
    "command_type": "HadoopCommand",
    "submitted_by": "demo2@qubole.com",
    "fs_bytes_read": 147972,
    "status": "done",
    "command_summary": "s3://paid-qubole/Examples/NewMaxTemperature.jar NewMaxTemperature \ns3n://"
  }
],
"start_date": "2014-04-01T10:00:00Z"
}

```

Note: We started collecting the CPU metrics only from the last week of December, 2013.

4.10.2 Canonical Hive Commands Report

GET /api/v1.2/reports/canonical/_hive/_commands/

This API provides you the canonical Hive commands report in JSON format.

Note: The following points are related to a report API:

- If the difference between start date and end date is more than 60 days, then the system defaults to 1 month window from the current day's date.
- If either start date or end date is not provided, then the system defaults to 1 month window from the current day's date.
- If you want to get data for a window more than 2 months, then write an email to solutions@quobole.com.

Parameters

Parameter	Description
start_date	The date from which you want the report. The report contains data from this date also. The API default is 30 days before the end_date. This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
end_date	The date until which you want the report. The report contains data from this date also. The API default is <i>today</i> or <i>now</i> . This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
offset	The starting point of the results. The API default is <i>0</i> .
limit	The number of results to fetch. The API default is <i>10</i> .
sort_column	The column used to sort the report. Since this report returns the top <i>canonical_hive_commands</i> , the sort order is always descending. Valid choices are ‘frequency’, ‘cpu’, ‘fs_bytes_read’, and ‘fs_bytes_written’. The API default is, <i>frequency</i> .
show_ast	Also return the serialized AST corresponding to the canonical query. (Not returned by default.)

Response Parameters

Parameter	Description
start_date	The actual starting date of the report.
end_date	The actual ending date of the report.
sort_column	The sort column used.

An array of:

canonical_query_id	The ID of the canonical query.
canonical_query	The AST dump of the canonical query. (This is returned only when the <i>show_ast</i> parameter is passed.)
recent_example	The most recent example of this type of queries.
frequency	The number of queries of this type.
cpu	The total cumulative CPU, (in ms), consumed by these queries.
fs_bytes_read	The total bytes read by these queries.
fs_bytes_written	The total bytes written by these queries.

Examples

Without any parameters

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" "https://api.qubole.com/api/v1/report"
```

Sample Response

```
{
  "sort_column": "frequency",
  "canonical_queries": [
    {
      "canonical_query_id": "af09cd5799e52f450a87e236f453b864833afac97603409a17f3df4d010b1814",
      "fs_bytes_written": 0,
      "fs_bytes_read": 0,
      "frequency": 8800,
      "cpu": 0,
      "recent_example": "alter table demo_memetracker recover partitions"
    },
    {
      "canonical_query_id": "9548ac7ec7defe3c3251da2544ec545c9bd578cb308c6c3c1936e48df0bdfdb4",
      "fs_bytes_written": 0,
      "fs_bytes_read": 0,
      "frequency": 4547,
      "cpu": 0,
      "recent_example": "alter table daily_tick_data recover partitions"
    },
    {
      "canonical_query_id": "69dae07fc876927b9daba6279c962bc343131c08d8f9f98adfa0c05ef90b40a4",
      "fs_bytes_written": 0,
      "fs_bytes_read": 0,
      "frequency": 768,
      "cpu": 0,
      "recent_example": "show tables"
    }
  ]
}
```

```
{
  "canonical_query_id": "89720fe23d2a85ac217a3b230e992c45dd523b65e6d45863cc410f4b5e4795ea",
  "fs_bytes_written": 0,
  "fs_bytes_read": 0,
  "frequency": 53,
  "cpu": 0,
  "recent_example": "select * from `default`.`memetracker` limit 400"
},
{
  "canonical_query_id": "04bccd848172842c8fadd687aef72ac2161f72895dfd3c1d3c31a96411d34095",
  "fs_bytes_written": 0,
  "fs_bytes_read": 0,
  "frequency": 49,
  "cpu": 0,
  "recent_example": "select * from `default`.`30days_test` limit 1"
},
{
  "canonical_query_id": "9996d665cf077f60b1ee87d3b5b80cd65ce078f77935096441433628909b9ddb",
  "fs_bytes_written": 9,
  "fs_bytes_read": 28482500000,
  "frequency": 48,
  "cpu": 0,
  "recent_example": "select count(*) from memetracker"
},
{
  "canonical_query_id": "492ea35ff3d58d0d07e70bcc68ea33eadb7bb572f6fdf14f7220931cc94b1abc",
  "fs_bytes_written": 0,
  "fs_bytes_read": 0,
  "frequency": 37,
  "cpu": 0,
  "recent_example": "select * from `default`.`default_qubole_airline_origin_destination` limit 10"
},
{
  "canonical_query_id": "5e2bb3326c4cf2c5b669d60729c5697fb6fdef4f1e09be41e37f424eb96b0c74",
  "fs_bytes_written": 0,
  "fs_bytes_read": 0,
  "frequency": 30,
  "cpu": 0,
  "recent_example": "select * from default_qubole_memetracker limit 10;"
},
{
  "canonical_query_id": "aa1c7a294f6e18feec68175a643814f06e180f4ac5e62eb6b556d9bf72830bc2",
  "fs_bytes_written": 25326,
  "fs_bytes_read": 85944,
  "frequency": 22,
  "cpu": 0,
  "recent_example": "select * from test_csv limit 5;"
},
{
  "canonical_query_id": "2145af0ee70e1cd93c9901cd41dee8285faacaefe86e4a4f22880316cc4e63c3",
  "fs_bytes_written": 21050200,
  "fs_bytes_read": 321138000,
  "frequency": 21,
  "cpu": 0,
  "recent_example": "select * from demo_memetracker limit 100"
}
]
```

With a different sort column and limit and show_ast=true

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" "https://api.qubole.com/api/v1/queries?sort_column=cpu&limit=2&show_ast=true"
```

Sample Response

```
{  
    "sort_column": "cpu",  
    "canonical_queries": [  
        {  
            "canonical_query_id": "d9635e3ad5501c9ad47bb728c35b63e1b41f8c2ba0fb4f7533b9ab701ce503c4",  
            "canonical_query": "(null(TOK_QUERY(TOK_FROM(TOK_TABREF(TOK_TABNAME(lineitem)))))(TOK_INSERT(TOK_TABREF(TOK_TABNAME(lineitem))))",  
            "fs_bytes_written": 18,  
            "fs_bytes_read": 7726360000,  
            "frequency": 2,  
            "cpu": 423130,  
            "recent_example": "set fs.s3.inputpathprocessor=false;\nselect count(*) from lineitem;"  
        },  
        {  
            "canonical_query_id": "eala37cf6b4694293d15cadfaf4bbae2459f12475cd86ea90e6c4f8e31945bda",  
            "canonical_query": "(null(TOK_QUERY(TOK_FROM(TOK_TABREF(TOK_TABNAME(default_qubole_memetracker)))))(TOK_INSERT(TOK_TABREF(TOK_TABNAME(default_qubole_memetracker))))",  
            "fs_bytes_written": 108,  
            "fs_bytes_read": 54673600000,  
            "frequency": 20,  
            "cpu": 416200,  
            "recent_example": "SELECT count(*) FROM default_qubole_memetracker where month = '2008-08';"  
        }  
    ]  
}
```

For a specific time period

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" -H "Accept: application/json" "https://api.qubole.com/api/v1/queries?start_time=2014-01-01T00:00:00Z&end_time=2014-01-02T00:00:00Z&sort_column=cpu&limit=2&show_ast=true"
```

Sample Response

```
{  
    "canonical_queries": [  
        {  
            "canonical_query_id": "55ebb0cc47e0dc74c70245b026126bba191969dee1dc380a6f98698e6b194085",  
            "cpu": 75720,  
            "frequency": 1,  
            "recent_example": "select dt, count(*) from junk_temp \ngroup by dt order by dt\n\n",  
            "fs_bytes_read": 308582016,  
            "fs_bytes_written": 1514  
        },  
        {  
            "canonical_query_id": "1c421d2e65407650650cbc2ee80f9a59863875f52d1a9ddd5a051118678a3a6c",  
            "cpu": 34980,  
            "frequency": 1,  
            "recent_example": "select created_at from junk_temp \nwhere dt=2014-01-03 limit 10\n\n",  
            "fs_bytes_read": 308582016,  
            "fs_bytes_written": 0  
        }  
    ]  
}
```

```
],
  "start_date": "2014-03-31T10:00:00Z",
  "end_date": "2014-04-21T20:00:00Z",
  "sort_column": "fs_bytes_read"
}
```

To learn more about canonicalization of hive queries, see the [Blog post](#).

Caution: We started collecting the CPU metrics only from the last week of December, 2013. So, if you have some queries before that their CPU will be considered as 0.

4.10.3 Cluster Nodes Report

GET /api/v1.2/reports/cluster_nodes

This API provides the cluster nodes report in JSON format.

Note: The following points are related to a report API:

- If the difference between start date and end date is more than 60 days, then the system defaults to 1 month window from the current day's date.
 - If either start date or end date is not provided, then the system defaults to 1 month window from the current day's date.
 - If you want to get data for a window more than 2 months, then write an email to solutions@quoble.com.
-

Parameters

Pa- ramete- r	Description
start_date	The date from which you want the report (inclusive). The API default is 30 days before the end date. This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
end_date	The date until which you want the report (inclusive). The API default is <i>today</i> . This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.

Response Parameters

Parameter	Description
start_date	The starting date of the report.
end_date	The ending date of the report.

An array of:

role	The role of the instance (master or slave).
cluster_id	The id of the cluster
public_ip	The public hostname of the cluster node.
ec2_instance_id	The ec2 instance ID of the cluster node.
private_ip	The private hostname of the cluster node.
start_time	The time at which the cluster node was started.
end_time	The time at which the cluster node was terminated.

Examples

Goal

To get the default report.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/cluster_nodes"
```

Goal

To get the report for clusters online during a specific time period.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/cluster_nodes?start_date=2014-04-01&end_date=2014-04-21"
```

Sample Response

```
{
  "end_date": "2014-04-21T10:00:00Z",
  "cluster_nodes": [
    {
      "ec2_instance_id": "i-437ad9ac",
      "private_ip": "ip-10-40-7-209.ec2.internal",
      "start_time": "2015-02-12T06:59:42Z",
      "role": "master",
      "public_ip": "ec2-23-20-255-83.compute-1.amazonaws.com",
      "end_time": "2015-02-12T08:13:52Z",
      "cluster_id": 10268
    },
    {
      "ec2_instance_id": "i-887bd867",
      "private_ip": "ip-10-165-32-171.ec2.internal",
      "start_time": "2015-02-12T06:59:42Z",
      "role": "node0001",
      "public_ip": "ec2-54-144-51-140.compute-1.amazonaws.com",
      "end_time": "2015-02-12T08:13:52Z",
      "cluster_id": 10268
    }
  ],
  "start_date": "2014-04-01T05:00:00Z"
}
```

4.10.4 Cluster Usage Report

GET api/v1.2/reports/cluster_usage_report

This API provides you the cluster usage report in JSON format.

Note: The following points are related to a report API:

- If the difference between start date and end date is more than 60 days, then the system defaults to 1 month window from the current day's date.

- If either start date or end date is not provided, then the system defaults to 1 month window from the current day's date.
- If you want to get data for a window more than 2 months, then write an email to solutions@qubole.com.

Parameters

Note: Parameters marked in **bold** below are mandatory. Others are optional and have default values.

Parameter	Description
start_date	The date from which you want the report (inclusive). The API default is 30 days before the end date. This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
end_date	The date until which you want the report (inclusive). The API default is <i>today</i> . This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
cluster_filter[]	Denotes the cluster ID of the cluster for which the usage report must be filtered. For example, cluster_filter[]=1 indicates that the usage report must be filtered to the cluster assigned with an ID 1.

Response Parameters

Parameter	Description
start_date	The starting date of the report.
end_date	The ending date of the report.
cluster_usage	An array of properties as mentioned in <i>Cluster Usage Properties</i> .

Cluster Usage Properties

qcuh	Displays the QCUH information
start_time	The time at which the cluster node was started.
end_time	The time at which the cluster node was terminated.
cluster_instance_id	The cluster instance ID of the cluster node.
cluster_id	The ID of the cluster
terminate_reason	The reason for the cluster to be terminated.
tags	Displays cluster labels
spot_nodes	Number of spot nodes spawned by the cluster
on_demand_nodes	Number of on-demand nodes spawned by the cluster

Examples

Request without a Cluster Filter

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN"
-H "Accept: application/json"
"https://api.qubole.com/api/v1.2/reports/cluster_usage_report?start_date=2015-08-10+18%3A30%3A00&end_
```

Request with a Cluster Filter

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN"
-H "Accept: application/json"
"https://api.qubole.com/api/v1.2/reports/cluster_usage_report?start_date=2015-08-10T18:30:00Z&end_
```

Response

```
{
  "cluster_usage": [
    {
      "qcuh": 3.0,
      "start_time": "2015-08-11 04:11:19 +0000",
      "end_time": "2015-08-11 05:01:15 +0000",
      "cluster_inst_id": 69759,
      "cluster_id": 1,
      "terminate_reason": "INACTIVITY",
      "tags": "default",
      "spot_nodes": 0.0,
      "on_demand_nodes": 2.0
    },
    {
      "qcuh": 3.0,
      "start_time": "2015-08-11 00:20:38 +0000",
      "end_time": "2015-08-11 01:15:18 +0000",
      "cluster_inst_id": 69700,
      "cluster_id": 1,
      "terminate_reason": "INACTIVITY",
      "tags": "default",
      "spot_nodes": 0.0,
      "on_demand_nodes": 2.0
    }
  ],
  "end_date": "2015-08-11T18:29:59Z",
  "start_date": "2015-08-10T18:30:00Z"
}
```

4.10.5 Tables Report

GET /api/v1.2/reports/tables_usage_frequency/

This API provides you the table frequency report, which contains a table name and the number of queries that have used it.

Note: The following points are related to a report API:

- If the difference between start date and end date is more than 60 days, then the system defaults to 1 month window from the current day's date.
 - If either start date or end date is not provided, then the system defaults to 1 month window from the current day's date.
 - If you want to get data for a window more than 2 months, then write an email to solutions@quoble.com.
-

Parameters

Pa- ram- eter	Description
start_date	The date from which you want the report (inclusive). The API default is 30 days before the end date. This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.
end_date	The date until which you want the report (inclusive). The API default is <i>today</i> . This parameter also supports timestamp in the UTC timezone (YYYY-MM-DDTHH:MM:SSZ) format.

Response Parameters

Parameter	Description
table	The name of the table.
frequency	Number of queries that refer to the table.

Examples

Goal

To get a default report.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/tables_usage_frequency"
```

Goal

To get a table usage frequency report for a specific time period.

```
curl -i -X GET -H "X-AUTH-TOKEN: $AUTH_TOKEN" \
-H "Accept: application/json" \
"https://api.qubole.com/api/v1.2/reports/tables_usage_frequency?start_date=2015-03-01&end_date=2015-03-02"
```


Qubole FAQs

5.1 General Questions

5.1.1 What is the pricing model for Qubole?

Please see the *Qubole Billing Guide* for a description of our pricing model.

Updated pricing plans are available at the [Pricing](#) page.

We also offer custom plans according to the customer requirements. For more information on the contact details, see the [Contact US](#) page.

5.1.2 What policy should I use for Qubole to use my IAM credentials?

If you are using IAM credentials for a Qubole account, you need (at a high-level) the following privileges:

- Launch and terminate machines
 - Create and delete security groups
 - Read access to data you wish to query
 - Read/Write access to a S3 bucket to store Hive tables, results, and logs

Sample IAM templates for EC2 settings and S3 policy are provided below.

Here is a sample IAM template for EC2 settings. (Note that you need to mention your bucket name here.)

```
"ec2:ModifyInstanceAttribute",
"ec2:RequestSpotInstances",
"ec2:RunInstances",
"ec2:StartInstances",
"ec2:StopInstances",
"ec2:TerminateInstances"
],
"Resource": [
"*"
]
}
]
}
```

Here is a sample IAM template for an S3 policy. (Note that you need to mention your bucket name here.)

```
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "Stmt1384922756000",
"Effect": "Allow",
"Action": [
"s3:DeleteObject",
"s3:GetObject",
"s3:GetObjectAcl",
"s3:PutObject",
"s3:PutObjectAcl"
],
"Resource": [
"arn:aws:s3:::<your-bucket1>/",
"arn:aws:s3:::<your-bucket1>/*"
]
},
"Resource": [
"arn:aws:s3:::<your-bucket1>/*",
"arn:aws:s3:::<your-bucket1>/*",
...
]
},
{
"Sid": "Stmt1384922862000",
"Effect": "Allow",
"Action": [
"s3:GetBucketAcl",
"s3>ListBucket"
],
"Resource": [
"arn:aws:s3:::<your-bucket1/*>",
"arn:aws:s3:::<your-bucket1/*>"
]
},
"Resource": [
"arn:aws:s3:::<your-bucket1/*>",
"arn:aws:s3:::<your-bucket1/*>",
...
]
},
{
"Sid": "Stmt1384923783000",
"Effect": "Allow",
```

```

"Action": [
  "s3:GetObject",
  "s3>ListBucket",
  "s3>ListAllMyBuckets",
  "s3:GetBucketLocation"
],
"Resource": [
  "arn:aws:s3:::paid-qubole/*",
  "arn:aws:s3:::public-qubole/*",
  "arn:aws:s3:::paid-qubole",
  "arn:aws:s3:::public-qubole",
  "arn:aws:s3:::*"
]
}
]
}
}

```

5.1.3 How does Qubole access data in my buckets?

Qubole accesses data using storage credentials added to the account's configuration. In addition to this, Qubole accesses data in the following ways:

- For Hive queries, Pig scripts, Hadoop jobs, and Presto queries, Qubole runs a Hadoop cluster on the machines that are rented by **your account**. The hadoop cluster reads, processes data, and writes the results back to your buckets. All the data is accessed on your machines.
- When you browse or download results from Qubole's website (UI or the API), the machines owned by Qubole will read the results from your bucket and provide them to you.
- For Data Import or Export commands, the data is transferred by a machine that runs within Qubole's account. For these commands, data in your buckets are accessed by Qubole's machines.

5.1.4 What is an invocation?

Qubole supports import and export of data from various external data sources into QDS and from QDS to various external data sources. These are named as Data Export and Data Imports commands. These commands are collectively called as **Invocations**.

5.1.5 What AWS regions are supported by Qubole?

Qubole currently supports following AWS regions:

- US East (N.Virginia) [us-east-1]
- US West (Seattle) [us-west-1]
- US West (Oregon) [us-west-2]
- EU (Ireland) [eu-west-1]
- Asia Pacific (Singapore) [ap-southeast-1]
- Asia Pacific (Tokyo) [ap-northeast-1]

5.1.6 Do files from an Hadoop archive get extracted to a specific folder by default?

When you extract files from an Hadoop archive, by default, files get extracted to a folder with a name that matches exactly with the Hadoop archive's name. For example, files from the `user/zoo/test.tar` archive are extracted into the `user/zoo/test.tar` folder.

5.2 Questions about Hive

5.2.1 What version of Hive does Qubole provide?

Qubole currently offers Hive-0.13. Qubole upgrades its Hive version frequently to offer latest stable releases from Apache.

In addition to stock functionality from Apache, Qubole adds a number of additional enhancements that include:

- Faster access to Data in S3
- Ability to recover partitions from S3
- JVM reuse across Hive queries to allow faster and more efficient query execution
- Ability to query MongoDB from Hive directly
- A responsive always-on Hive server for fast interactive queries
- Automatic caches HDFS to speed up access to json data

Qubole's Hive service also benefits from a highly optimized and tuned deployment of Hadoop that provides faster and more efficient job scheduling as well as auto-scaling benefits.

5.2.2 My data is in a directory structure in S3. How can I access it?

Map-Reduce and related frameworks like Hive, Pig and Presto can process files in S3 easily.

To analyze data in S3 using Hive – we need to define a Hive table over S3 directories. This can be done a Hive DDL statement, for example:

```
CREATE EXTERNAL TABLE myTable (key STRING, value INT) LOCATION 's3n://mys3bucket/myDir';
```

where `myDir` is a directory in S3 in the bucket `mys3bucket`. If `myDir` has subdirectories – then the Hive table must be declared to be a partitioned table with a partition corresponding to each subdirectory. In addition Qubole makes it easy to explore S3 and define Hive tables over the same using the [Data Wizard](#).

For Map-Reduce jobs, like those launched using Hadoop Streaming we can input directories via options in command line.

5.2.3 What is the difference between an external table and a managed table?

The main difference is that when you drop an external table, the underlying data files stay intact. This is because the user is expected to manage the data files and directories. With a managed table, the underlying directories and data get wiped out when the table is dropped.

5.2.4 How can I create a table in HDFS?

A *CREATE TABLE* statement in Qubole creates a managed table in S3. If you want to create a table holding intermediate data in HDFS, you can use *CREATE TMP TABLE*. Note that HDFS in Qubole is ephemeral and gets wiped out when the cluster is shutdown. Use HDFS only for producing intermediate outputs. *CREATE TMP TABLE* is Qubole's custom extension and is not part of Apache Hive.

5.2.5 What file formats does Qubole's Hive support out of the box?

Qubole's Hive supports:

- Text files (compressed or uncompressed) containing delimited, CSV or JSON data.
- It can support Binary data files stored in RCFile and SequenceFile formats containing data serialized in Binary JSON, Avro, ProtoBuf and other binary formats

Custom File Formats (InputFormats) and Deserialization libraries (SerDe) can be added to Qubole. Please contact [Qubole Support](#) if you have a query in this regard.

5.2.6 How do I disable/enable a feature in Hive for my workload?

Coming Soon...

5.2.7 What is the default InputFormat used by Qubole's Hive?

Qubole's hive uses *CombineHiveInputFormat* by default and treats underlying files as text-files by default. During table definition – users can indicate that the file type of of an alternative format (such as a Sequence or RC File).

5.2.8 Does Qubole remember my tables even when my cluster goes away?

For every account in Qubole – we create a persistent MySql backed Hive metastore automatically. All table definitions are stored in this metastore – and hence table metadata is available across cluster restarts.

(Note that *tmp* tables are an exception as they are automatically deleted at the end of a user session)

5.2.9 I have my data in RDS. Can I use Hive to process the data?

Yes. You can use “Data Import” command to fetch data from RDS and many other non rdbms databases in a hive table. Then normal hive queries can be run on the hive table to do processing.

As a final step you can run “Data Export” command on the processed data to export it back to a database of your choice for visualization purposes.

5.2.10 Can I use Excel/Tableau/BI tools on top of Qubole's Hive tables?

Qubole has an ODBC driver (in Beta, currently) that allows Excel, Tableau and other BI tools to talk to Qubole's Hive service. Please contact [Qubole Support](#) to request access.

5.2.11 Can I plug in my own UDFs and SerDes?

Yes you can! However because Qubole is a multi-tenant service – we manually approve such requests on a per-account basis. Please contact help@qubole.com if there is a data format that is not supported or if you wish to use custom user-defined functions (UDF).

5.3 Questions about Hadoop Clusters

5.3.1 How long does a Qubole Hadoop Cluster take to come up?

It takes about 90s for Qubole to spin up a Hadoop Cluster. We have pre-baked images which has all required hadoop and related packages installed. We configure all the dynamic stuff like user credentials at runtime using custom scripts which run when we bring up the node. The cluster bring up time does not actually increase with the number of nodes. We can bring up 100 or 1000 node hadoop cluster in approximately 90 seconds.

5.3.2 On whose AWS account are the hadoop clusters launched?

Hadoop clusters are launched on customers AWS account. While creating an account, Qubole take your AWS credentials for storage and compute which are used to spin up EC2 instances and talk to your s3 buckets.

5.3.3 When are clusters brought up and shutdown?

Qubole's hadoop clusters are ephemeral. Since the actual data lives in object store like s3/gcs, we dont need to keep the cluster alive and can tear it down.

Once a query is submitted through UI/API/SDK – say a hive command or any command which runs Map-Reduce job, we check if there is an active cluster running for this account and bring up cluster automatically if one does not exist. For the subsequent queries, if the cluster is already active, it will continue using the same cluster. If the cluster is idle for sometime due to inactivity, we bring down the cluster.

For hive metadata queries like “show table” does not actually need a hadoop cluster. For these type of queries, it doesnt bring up a cluster if it did not exist before.

Also, user could manually bring up or bring down the cluster. Go to Analyze UI => cluster tab => cluster icon to perform this.

Note : We also have a feature to disable auto-termination of the cluster. Check control panel in Qubole UI for this option. Make sure that if this option is selected, you manually terminate the cluster using the cluster tab in Analyze UI.

5.3.4 When are clusters auto-scaled?

Qubole monitors the load on a running cluster and will scale it up if the running Hadoop jobs can be speeded up by adding more machines. Similarly, Qubole will remove nodes when they are no longer required. When removing nodes, we make sure they are never released before the full hour is up – so you get your money's worth.

5.3.5 Should I have one large auto-scaling cluster or multiple smaller clusters?

This is a common question that we come across while using QDS! Note that all clusters under an account share the Hive metastore and Storage Credentials used to access data in S3. So, the same set of commands/queries can be run irrespective of a cluster configuration. Some trade-offs are listed below:

- Sharing a single large auto-scaling cluster allows efficient usage of compute resources. Consider a scenario:
 - An application, *App1*, provisions a cluster but finishes leaving 30 minutes on the clock before the cluster reaches the hour mark and is terminated.
 - Another application, *App2*, now needs to be run. If *App2* uses the same cluster, then it can use paid-for compute resources that would otherwise go waste.

So, sharing clusters can lower the cost. Lower cost is interchangeable with higher performance. You can spin-up larger clusters for the same cost. A diligent Fair-Scheduler configuration on a shared cluster can provide responsive behavior even with multiple users.

- Multiple clusters allow configurations optimized for different workloads

This is part of the reason why QDS has different types of clusters for different distributed runtime (such as Hadoop/Spark). Memory-intensive applications benefits from high memory instances (such as the *r3* types in AWS) while compute-intensive benefits from a different instance type (such as *c3* types in AWS).

Multiple clusters are also required if, for example, different data sets are resident in different regions. It is better in that case to run multiple clusters, each closer to where the data resides.

- Multiple clusters leads to higher isolation

Although QDS uses frameworks such as the Hadoop Fair-Scheduler to arbitrate access to a common cluster, contention can be avoided with the use of multiple clusters. This is an issue if there are production jobs with stringent SLAs. Running them on a separate cluster is always safe but expensive.

- Efficiency gains from a shared cluster depend on type of job

For example, a small job that runs every 15 minutes does not see much gain by sharing compute resources with larger bursty jobs. As such jobs are also often SLA-driven, it is better to run them on a different cluster.

5.3.6 Will HDFS be affected by cluster auto-scaling?

No. Qubole always removes nodes from HDFS gracefully before terminating them. This allows HDFS to safely replicate data to surviving nodes of a cluster and as a result data stored in HDFS is not lost. However HDFS lasts only for the lifetime of a cluster and its contents are lost when the cluster is automatically terminated. Hence we recommend using HDFS only as a temporary data store for data flowing between jobs/queries.

5.3.7 Can I use AWS Reserved Instances with Qubole?

Yes, AWS Reserved Instances can be used by selecting the AWS region and Availability Zone for Hadoop cluster in which the instances have been reserved.

5.3.8 Can I use AWS Spot instances with Qubole?

Yes, Spot Instances can be used with Qubole. Using the **Slave node purchasing option** in cluster configuration, various strategies can be used for adding nodes to the cluster depending on the cost and availability of Spot Instances.

5.3.9 How can I use Reserved Instances with Qubole?

Qubole uses Linux/Unix instances as cluster nodes. You can use reserved instances with Qubole by purchasing Reserved Instances (RIs) from Amazon of type Linux/Unix to reduce your long running cluster costs. The instance size you reserve must be the same as the instance type you specify while configuring clusters. The RIs must be in VPC/non-VPC depending on whether your cluster is in VPC or non-VPC.

5.3.10 What is the difference between purchasing option for slave nodes?

The purchase option for slave nodes is applicable to nodes which are dynamically added in auto-scaling Hadoop cluster.

When the cluster is up, it does not scale below the specified minimum slave count. The minimum set of nodes are always On-Demand Instances. When the cluster scales up, Spot Instances can be used to significantly reduce compute costs.

QDS supports three purchase options for auto-scaled slave nodes:

- **On-Demand Instance:** All instances in the cluster are On-Demand Instances. This option can be used when quick auto-scaling is required with maximum stability.
- **Spot Instance:** Instances other than minimum slave count are Spot Instances. Users can specify the bid percentage and timeout to wait for the Spot Instance. If the Spot Instance is not available, the cluster will not scale up. This option can be used for more compute power when cheap Spot Instances are available.
- **Hybrid:** This option provides a flexible way to control the composition of instances in the cluster. Users can specify an upper limit on the percentage of Spot Instances. Under this option, if the Spot Instances are available then they can be added to the cluster conforming to the percentage specified. In case Spot Instances are not available, On-Demand Instances will be used for scaling up. Using this option, a replica of HDFS data block on Spot Instance is also kept on a stable On-Demand Instance to avoid data corruption in case all the Spot Instance are withdrawn (e.g. due to spike in Spot market). This option can be used for acquiring Spot Instances when available or fall back to On-Demand Instances in case of unavailability.

5.3.11 Does Qubole store any data?

As a trial user, you can use Qubole's compute and storage resources at no cost for a 15-day trial.

5.3.12 Can the data stored on EC2 instance be encrypted?

The data at rest on ephemeral drives of an EC2 instance can be kept encrypted for enhanced security. QDS Hadoop cluster uses ephemeral drives for HDFS and also for intermediate output of Map-Reduce application. If the option, **Enable encryption of instance local storage** is selected in the cluster configuration, then an encrypted file system is used for ephemeral drives.

5.3.13 Can I use Python 2.7 for Hadoop tasks?

On cluster nodes the default version of Python is 2.6. It is possible to use Python 2.7 for Hadoop tasks by adding the following lines to the node bootstrap file specified in the cluster configuration.

```
source /usr/lib/hustler/bin/qubole-bash-lib.sh  
qubole-hadoop-use-python2.7
```

The next invocation of the cluster will use Python 2.7 for Hadoop tasks.

5.4 Questions about Security

5.4.1 Where do my Hive queries/Hadoop jobs run?

All hadoop commands run on a hadoop cluster. Every hadoop command brings up a cluster.

Hive commands run on qubole machines, but they run MR jobs on a hadoop cluster. This makes sure that meta data queries like “show table”, “recover partitions” etc. don’t bring up a hadoop cluster. Thus saving money for customers.

5.4.2 Does Qubole store any of my data in its machines?

Qubole doesn’t store any of our users’ data in its machines. The clusters are always launched using user’s credentials in the user’s account on the cloud. The results of every single query executed by the user is also stored in the user’s account on the cloud. Our HDFS cache (which is ephemeral) which is used while running queries supports encrypted storage. We may temporarily cache some query results but they are also done on ephemeral machines. The user’s credentials are encrypted in our database.

/api

GET /api/v1.2/actions,	392	POST /api/v1.2/scheduler/,	374
GET /api/v1.2/actions/(int:id),	393	POST /api/v1.2/scheduler/(SchedulerID)/duplicate,	385
GET /api/v1.2/commands/(int:command_id),	296	POST /api/v1.3/clusters/,	332
		POST /api/v1.3/clusters/(string:id_or_label)/clone,	
GET /api/v1.2/commands/(int:command_id)/jobs,	299	POST /api/v1.3/clusters/(string:id_or_label)/nodes,	
		358	
GET /api/v1.2/commands/(int:command_id)/logs,	298	POST /api/v1.3/clusters/(string:id_or_label)/snapshot,	
		363	
GET /api/v1.2/commands/(int:command_id)/results,	297	PUT /api/v1.2/actions/(int:id)/kill,	391
		PUT /api/v1.2/actions/(int:id)/rerun,	
GET /api/v1.2/hive/default/,	324	391	
GET /api/v1.2/hive/default/**table**,	328	PUT /api/v1.2/clusters/,	369
		PUT /api/v1.2/commands/(int:id),	304
GET /api/v1.2/hive/default/*table*/properties,	330	PUT /api/v1.2/scheduler/(Scheduler	
		ID),	381
GET /api/v1.2/reports/all_commands,	395	PUT /api/v1.2/scheduler/(int:id),	384
GET /api/v1.2/reports/canonical/_hive/_commands,	397	PUT /api/v1.3/clusters/(string:id_or_label),	
		357	
GET /api/v1.2/reports/cluster_nodes,	401	PUT /api/v1.3/clusters/(string:id_or_label)/nodes,	
		360	
GET /api/v1.2/reports/tables_usage_frequency,	404	PUT /api/v1.3/clusters/(string:id_or_label)/restore,	
		364	
GET /api/v1.2/scheduler/,	379	PUT /api/v1.3/clusters/(string:id_or_label)/snapshot,	
		366	
GET /api/v1.2/scheduler/(int:id),	377	PUT /api/v1.3/clusters/(string:id_or_label)/state,	
		350	
GET /api/v1.2/scheduler/(int:id)/actions,	388	PUT /api/v1.3/clusters/reassign-label,	
		353	
GET /api/v1.3/clusters,	331	DELETE /api/v1.2/db_taps/(int:dbtap_id)/,	
		370	
GET /api/v1.3/clusters/(string:id_or_label),	351	DELETE /api/v1.2/hive/default/(string:table_name)/p	
		354	
GET /api/v1.3/clusters/(string:id_or_label)/metrics,	364	DELETE /api/v1.3/clusters/,	352
		330	
GET /api/v1.3/clusters/(string:id_or_label)/snapshot_schedule,	367	DELETE /api/v1.3/clusters/(string:id_or_label)/node	
		361	
GET /api/v1.2/reports/cluster_usage_report,	402		
POST /api/v1.2/commands/,	309		
POST /api/v1.2/hive/schema/table,	329		