Exercise 7.3 – Spark UI

The purpose of this exercise is to practice using Spark UI so you have a basic familiarity with the interface.

In this exercise, you will:

- Understand how to start the Spark master and application UIs
- Become familiar with the basic layout and use of each UI.

Overview

In this exercise you will start two *spark* clients. Then you will use a browser to view the Master UI (there will only be one) and the two application UIs (one for each client). From there you will submit jobs and observe how the UIs reflect what is happening with each job.

Part 1: Start the Spark SQL client

Starting the Spark SQL client is simple.

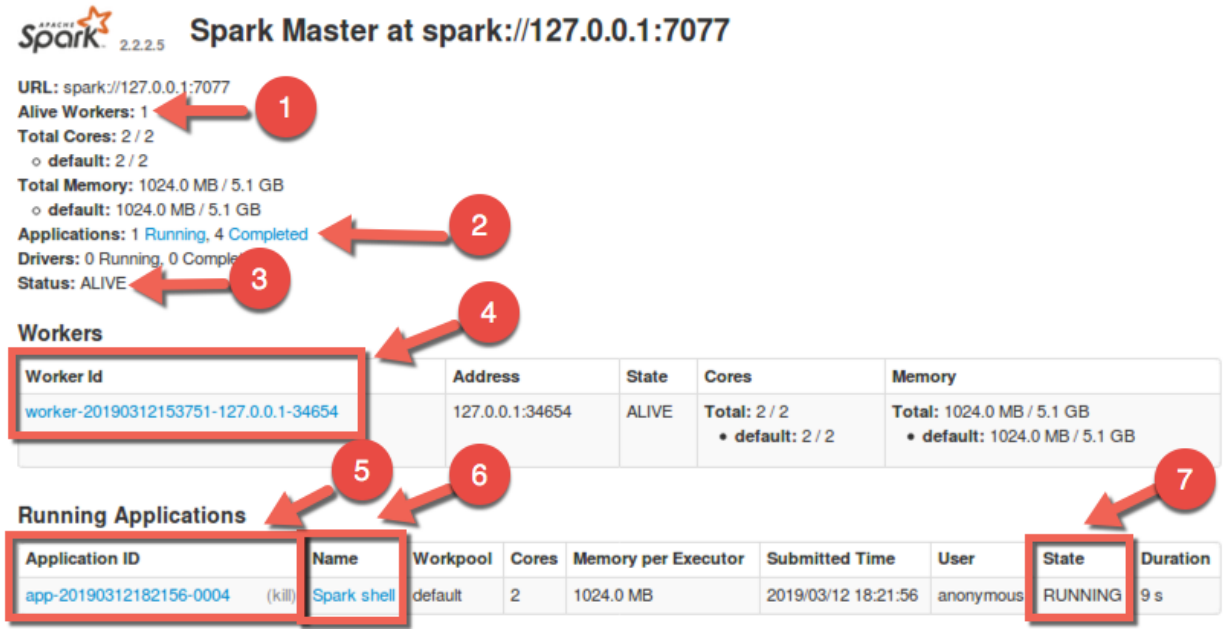
1. Open a terminal window.
2. Type `dse spark` and hit Enter.
3. After a brief delay, the *spark repl* command line should appear as follows:

```
Creating a new Spark Session
Spark context Web UI available at http://10.0.2.15:4040
Spark Context available as 'sc' (master = dse://?, app id = app-
   20190312162210-0001).
Spark Session available as 'spark'.
Spark SqlContext (Deprecated use Spark Session instead) available as
   'sqlContext'
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.2.2.5
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java
   1.8.0_161)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Part 2: Launch the Master UI

Launch the Master UI.

1. Open a browser window.
2. Type http://<NODE1-IP>:7080/ in the URL bar and hit enter.
3. You should see a screen similar to the following:

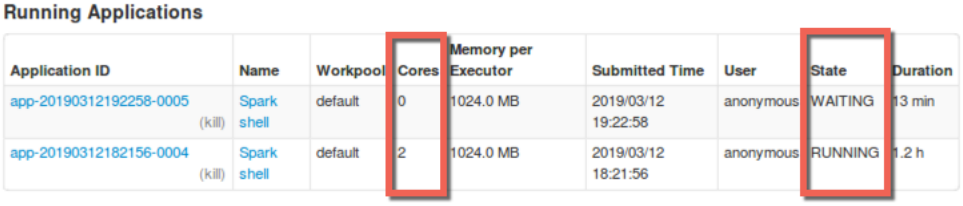


As you will recall from the lecture, the Master UI shows the state and health of the Spark Master. The master communicates both with clients and with the workers to make sure work gets done.

4. Using the screenshot as a guide, look at your own Master UI and answer the following questions:
   a. How many live workers do you have?
   b. How many applications are running?
   c. What is the status of the master UI?
   d. What is the ID of the worker(s)?
   e. Check that the application ID shown in the master UI under running applications matches that of the application ID in your Spark Shell (see terminal window)
   f. Verify that the application shown in the Master UI is in a running state

5. Now open a second terminal window, type `dse spark` and hit *Enter*.

6. Once the Spark REPL command prompt appears in the second terminal window, refresh the Master UI and check the number of running applications. There should be two. Also, the Application IDs should match those of each spark session.



```
Creating a new Spark Session
Spark context Web UI available at http://10.0.2.15:4040
Spark Context available as 'sc' (master = dse://?, app id = app-20190312182156-0004).
```

```
Spark context Web UI available at http://10.0.2.15:4041
Spark Context available as 'sc' (master = dse://?, app id = app-20190312192258-0005).
```

7. Note the “kill” link next to each application ID. You can kill applications as needed using that link.

8. Also notice that the newly launched application will probably be in a waiting state if the number of cores assigned to it is 0. This is because you are on a VM and have limited resources, and thus it has no cores to use.

Part 3: Launch the Application UIs

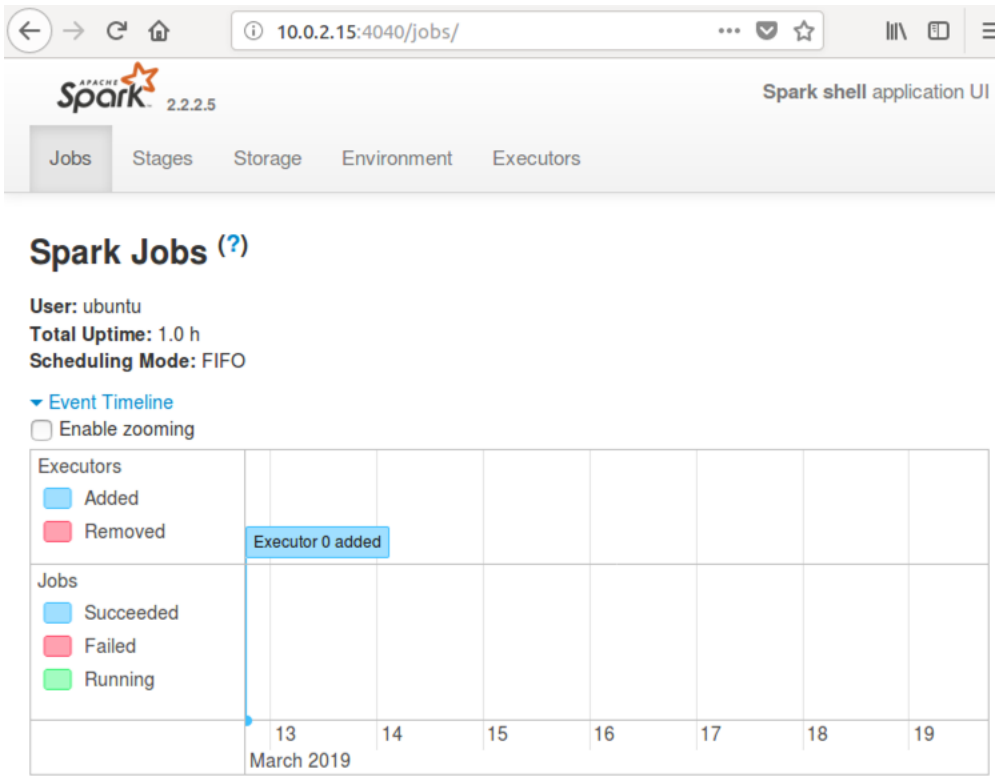
Launch the first Application UI.

1. Open a browser window.

2. Look at the first terminal window you opened and locate this line:

```
Spark context Web UI available at http://<NODE1-IP>:4040
```

3. The URL is for the Application UI for the Spark Session shown in that terminal window. Take the URL, open a new tab on the browser, paste the URL into the URL bar, and hit enter. The page that appears should appear as shown below:



4. Click on *Jobs and Stages* at the top of the page. Notice that there is no useful information other than the creation of Executor 0 shown under Spark Jobs. This is because no work is currently being done.

5. Give the Spark REPL a job. Cut and paste the following code into the command line of the first terminal window and hit enter:

```
val results = spark.sql("SELECT * FROM killrvideo_spark.users LIMIT
   10")
```

6. You should see the following once the code executes:

```
results: org.apache.spark.sql.DataFrame = [userid: string,
   created_date: timestamp ... 3 more fields]
```

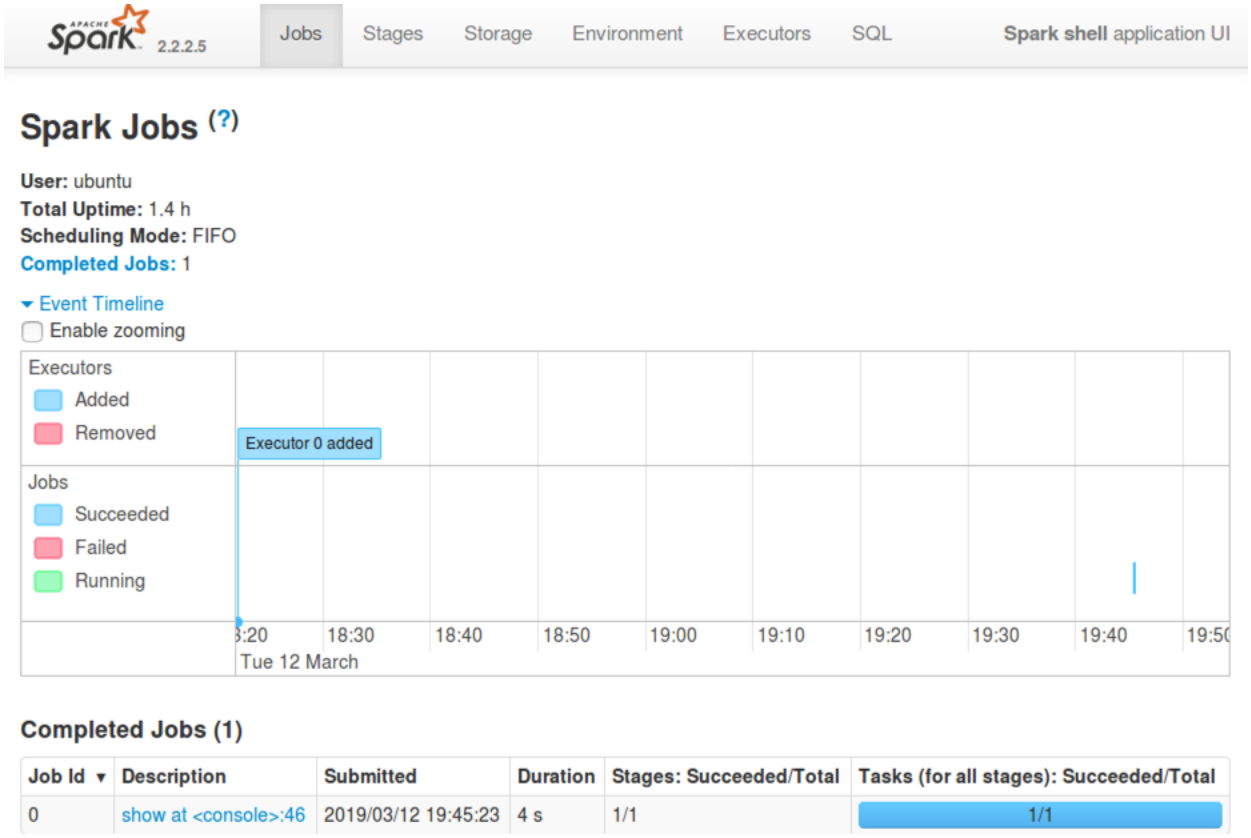
7. Now go back to the application UI and refresh the page. Click on Jobs and Stages at the top of the page. You will notice that nothing has changed. This is because all we have done is create a dataset. Now we need to get it to actually run the query and gather the data we need. Go back to the terminal window, type `results.show()` and hit Enter.

8. You should now see this in the terminal window:

```
+--------------------+------------------+--------------------+---------+---------+
|              userid|      created_date|               email|firstname| lastname|
+--------------------+------------------+--------------------+---------+---------+
|6d98290e-39ec-44e...|2013-06-25 00:00:00|widdendencr@googl...|  Whitney| Iddenden|
```

```
|4cd3bf7f-52cb-470...|2012-02-09 00:00:00|htargettpc@squido...|  Harwell|  Targett|
|b73c452a-cd06-4aa...|2014-03-12 00:00:00| kgallehawkjy@ed.gov|    Karyl|Gallehawk|
|3d2711d6-2843-4a5...|2014-10-15 00:00:00| rleopardg@weibo.com|  Reynold|  Leopard|
|3577ed76-6081-43e...|2010-04-07 00:00:00|lmcelreeh5@siteme...| Leonidas|  McElree|
|1baa092f-9610-412...|2010-06-04 00:00:00|tfurmengerax@jiat...|   Tobias|Furmenger|
|7608a5a0-65b3-4e3...|2016-01-05 00:00:00|  ywarrepg@yandex.ru|     Yuri|    Warre|
|fd0ec920-2829-47f...|2013-06-02 00:00:00|rjirieckr6@bibleg...|   Rossie|  Jirieck|
|e198a897-869d-4fa...|2015-01-27 00:00:00| fleemingcr@narod.ru|  Filbert|  Leeming|
|c0590cee-0db2-4fe...|2013-08-13 00:00:00|dmulvaneybp@youtu.be|   Darbie| Mulvaney|
+--------------------+-------------------+--------------------+---------+---------+
```

9. Now go to the Application UI and refresh the screen. You should see the following new information:



As you can see there is a completed job.

10. Now take the Application UI URL of the second terminal window and launch it in another browser tab. Note that its port number will be +1 of the previous port number. That is because if the default port 4040 is taken (as it was because the previous Application UI was using it) Spark keeps trying each subsequent port number until it finds an available one to use.

11. Now run the same query in the REPL window.

```
val results = spark.sql("SELECT * FROM killrvideo_spark.users LIMIT
  10")
```

12. You should see the following once the code executes.

```
results: org.apache.spark.sql.DataFrame = [userid: string,
   created_date: timestamp ... 3 more fields]
```

13. If you check the Application UI it should show no useful information as before. Now execute this code:

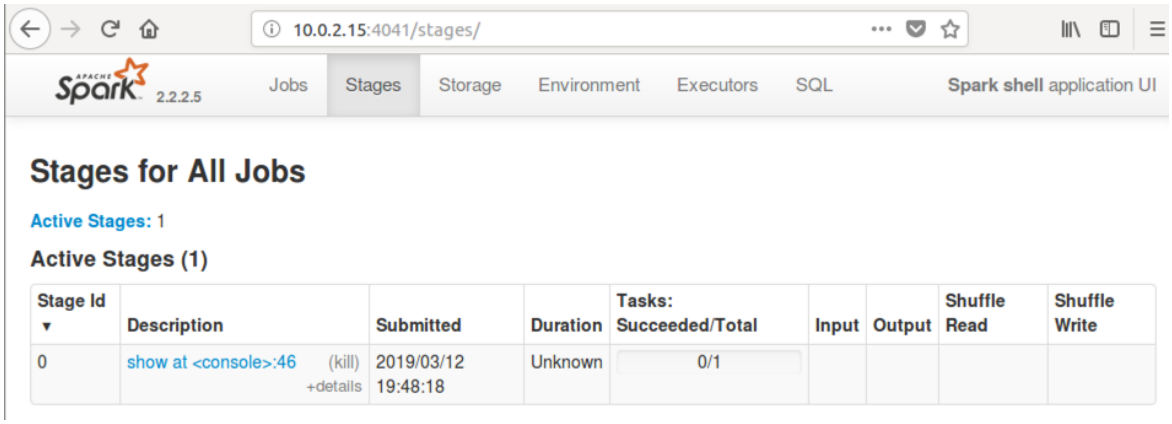
```
scala> results.show()
```

14. If this application showed a status of “waiting” in the Master UI window, you will probably get the following results in the terminal window and the Stages tab of the Application UI, respectively:

    Terminal Window:

```
WARN  2019-03-12 19:48:33,690
   org.apache.spark.scheduler.TaskSchedulerImpl: Initial job has not
   accepted any resources; check your cluster UI to ensure that workers
   are registered and have sufficient resources
```

    Stages Tab:



15. This is because this application has no resources with which to execute the job. However, if the application did have resources and the job executes, you’ll see a result identical to what you saw with the first application and you can skip the next step.

16. Go back and terminate the job in the terminal window using `CTRL+C`.