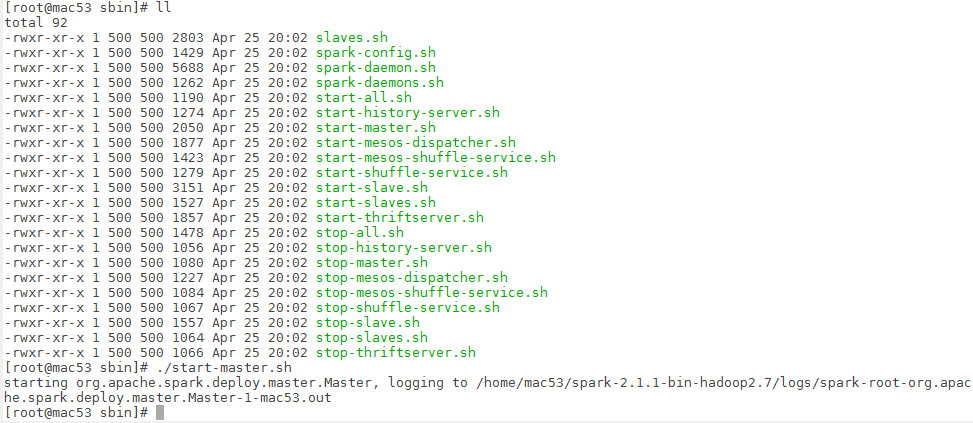
# Standalone mode

Here I want to create spark cluster using two nodes , I have downloaded spark and extract it on mac53 and mac55 machines

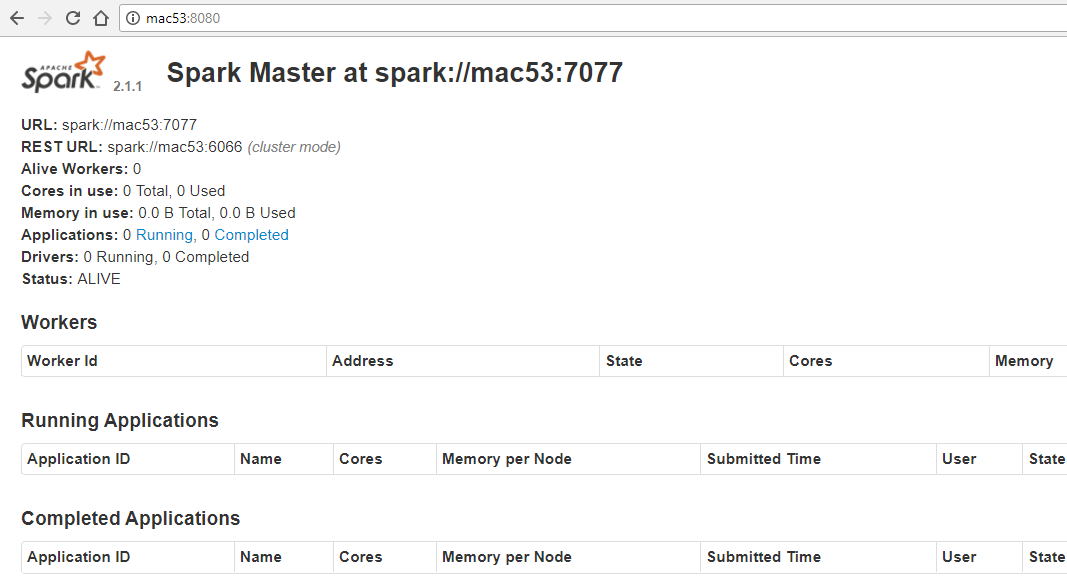
## Start Cluster by running individual shell files

### On mac53 machine

Go to sbin folder and run spark-master.sh file



Now access <http://host:8080> to get spark master URL



Now you can run slave using spark master command **./sbin/spark-slave.sh spark://mac53:7077**



### On mac55 machine

Start slave on mac55 also

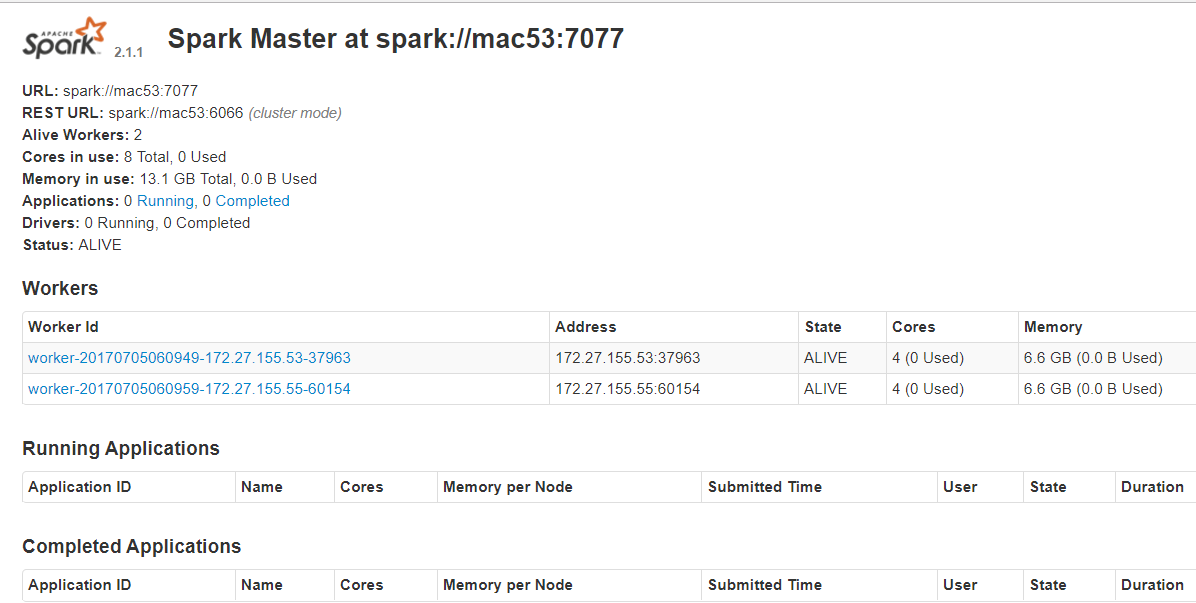
|  |
| --- |
| [root@mac55 sbin]# ./start-slave.sh spark://mac53:7077  starting org.apache.spark.deploy.worker.Worker, logging to /home/mac55/spark-2.1.1-bin-hadoop2.7/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-mac55.out |

Now we can run spark shell using below command

|  |
| --- |
| .bin/spark-shell spark://mac53:7077 |

|  |
| --- |
| [root@mac55 sbin]# cd ../bin  [root@mac55 bin]# ./spark-shell spark://mac53:7077  Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties  Setting default log level to "WARN".  To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  17/07/05 06:02:16 WARN SparkContext: Support for Java 7 is deprecated as of Spark 2.0.0  17/07/05 06:02:16 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  17/07/05 06:02:24 WARN ObjectStore: Failed to get database global\_temp, returning NoSuchObjectException  Spark context Web UI available at http://172.27.155.55:4040  Spark context available as 'sc' (master = local[\*], app id = local-1499248937502).  Spark session available as 'spark'.  Welcome to  \_\_\_\_ \_\_  / \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_  \_\ \/ \_ \/ \_ `/ \_\_/ '\_/  /\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 2.1.1  /\_/  Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.7.0\_51)  Type in expressions to have them evaluated.  Type :help for more information.  scala> |

Now check Web UI again



Here we can see two worker nodes are created, one on mac53 and other on mac55

By default, the cluster manager will automatically allocate the amount of CPUs and memory on each worker and pick a suitable default to use for Spark.

To submit the job in standalone mode use below command

|  |
| --- |
| spark-submit --class "org.cyb.spark.simpleSpark" **--master spark://mac53:7077** /home/mac55/Spark211-0.0.1-SNAPSHOT.jar |

Finally, the Standalone cluster manager supports two deploy modes for where the driver program of your application runs. In client mode (the default), the driver runs on the machine where you executed spark-submit, as part of the spark-submit command.

This means that you can directly see the output of your driver program, or send input to it (e.g., for an interactive shell), but it requires the machine from which your application was submitted to have fast connectivity to the workers and to stay available for the duration of your application. In contrast, in cluster mode, the driver is launched within the Standalone cluster, as another process on one of the worker nodes, and then it connects back to request executors. In this mode spark-submit is “fire-and-forget” in that you can close your laptop while the application is running.

You will still be able to access logs for the application through the cluster manager’s web UI. You can switch to cluster mode by passing --deploy-mode cluster to spark-submit.

You can also launch spark-shell or pyspark against the cluster in the same way, by passing the --master parameter:

|  |
| --- |
| spark-shell --master spark://masternode:7077  pyspark --master spark://masternode:7077 |

In the Standalone cluster manager, resource allocation is controlled by two settings:

**Executor memory**

You can configure this using the **--executor-memory** argument to spark-submit. Each application will have at most one executor on each worker, so this setting controls how much of that worker’s memory the application will claim. By default, this setting is 1 GB—you will likely want to increase it on most servers.

**The maximum number of total cores**

This is the total number of cores used across all executors for an application. By default, this is unlimited; that is, the application will launch executors on every available node in the cluster. For a multiuser workload, you should instead ask users to cap their usage. You can set this value through the

--**total-executorcores** arguments to **spark-submit**, or by configuring spark.cores.max in your Spark configuration file.

To stop slave and master use below command

|  |
| --- |
| [root@mac53 sbin]# ./stop-slave.sh  stopping org.apache.spark.deploy.worker.Worker  [root@mac53 sbin]# ./stop-master.sh  stopping org.apache.spark.deploy.master.Master |

## Start all workers at once

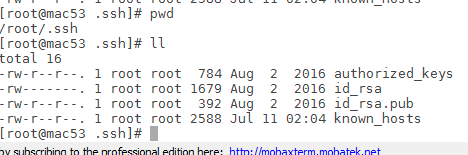
### Setup password less connection

Here mac53 is our master machine and we want to run slaves on mac53 and mac55

First we need to setup password less connection between mac53 and mac55

On mac53 run below command

|  |
| --- |
| ssh-keygen -t rsa   * It cerate id\_rsa file under .ssh directory for private key * And id\_rsa.pub file for public key |



Now copy the content of id\_rsa.pub file to authourized\_keys file that will be present under mac55 machine (~/.ssh folder)

Run below command on mac53 machine to copy the content of public key to mac55 machine

|  |
| --- |
| [root@mac53 .ssh]# ssh-copy-id -i ~/.ssh/id\_rsa.pub root@mac55  /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed  /usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys  root@mac55's password:  Number of key(s) added: 1  Now try logging into the machine, with: "ssh 'root@mac55'"  and check to make sure that only the key(s) you wanted were added. |

Now again run the copy command to copy the content of public key to mac53 machine(it own authorized\_keys file)

|  |
| --- |
| [root@mac53 .ssh]# ssh-copy-id -i ~/.ssh/id\_rsa.pub root@mac53  /usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed  /usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys  root@mac53's password:  Number of key(s) added: 1  Now try logging into the machine, with: "ssh 'root@mac53'"  and check to make sure that only the key(s) you wanted were added. |

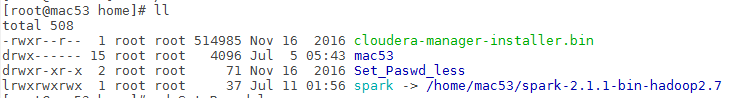
Now password less connection is set between mac53 and mac55 (master to slave)

### Setup spark on same directory structure

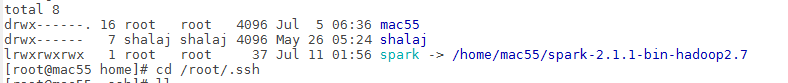
Copy a compiled version of Spark to the same location on all your machines—for example, /home/yourname/spark

As we already setup spark on different directories we can create softlinks on same location

|  |
| --- |
| ln -s /home/mac53/spark-2.1.1-bin-hadoop2.7 /home/spark |



|  |
| --- |
| ln -s /home/mac55/spark-2.1.1-bin-hadoop2.7 /home/spark |



### Start master and slaves

First go to <spark\_home>/conf folder and copy slaves.template

|  |
| --- |
| #cp slaves.template slaves |

Now edit slaves file and add all workers node host name

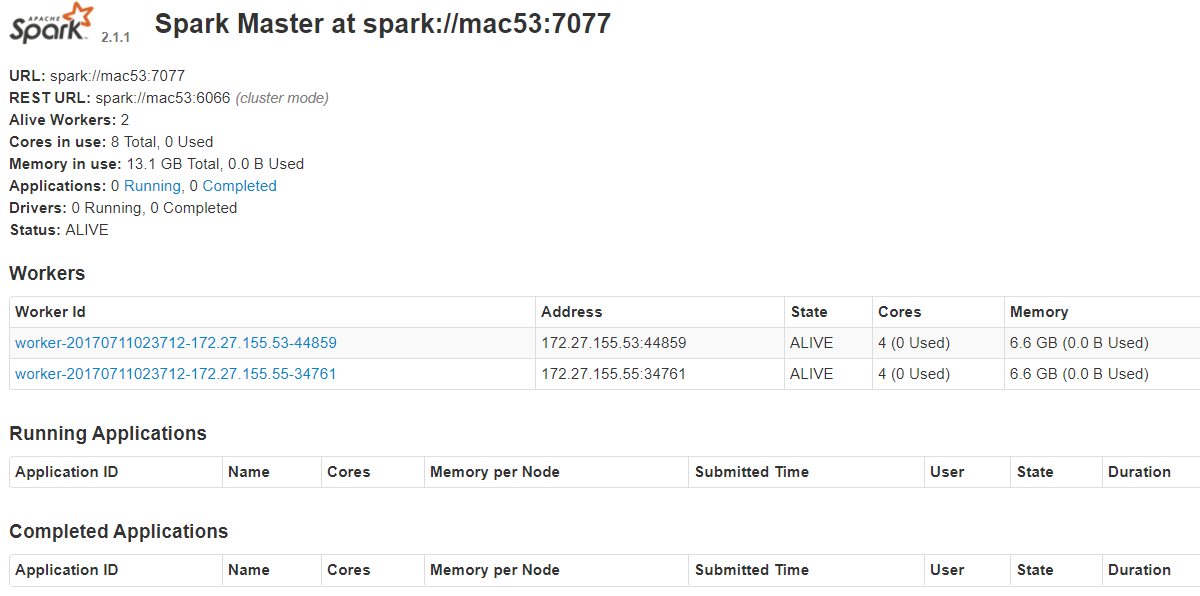
|  |
| --- |
| #  # Licensed to the Apache Software Foundation (ASF) under one or more  # contributor license agreements. See the NOTICE file distributed with  # this work for additional information regarding copyright ownership.  # The ASF licenses this file to You under the Apache License, Version 2.0  # (the "License"); you may not use this file except in compliance with  # the License. You may obtain a copy of the License at  #  # http://www.apache.org/licenses/LICENSE-2.0  #  # Unless required by applicable law or agreed to in writing, software  # distributed under the License is distributed on an "AS IS" BASIS,  # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  # See the License for the specific language governing permissions and  # limitations under the License.  #  # A Spark Worker will be started on each of the machines listed below.  mac53  mac55 |

Now go to sbin folder on master node (mac53) and run start-all.sh file

|  |
| --- |
| [root@mac53 spark]# cd /home/spark/sbin  [root@mac53 sbin]# **./start-all.sh**  starting org.apache.spark.deploy.master.Master, logging to /home/spark/logs/spark-root-org.apache.spark.deploy.master.Master-1-mac53.out  mac53: starting org.apache.spark.deploy.worker.Worker, logging to /home/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-mac53.out  mac55: starting org.apache.spark.deploy.worker.Worker, logging to /home/spark/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-mac55.out |

Now check url below

<http://mac53:8080>



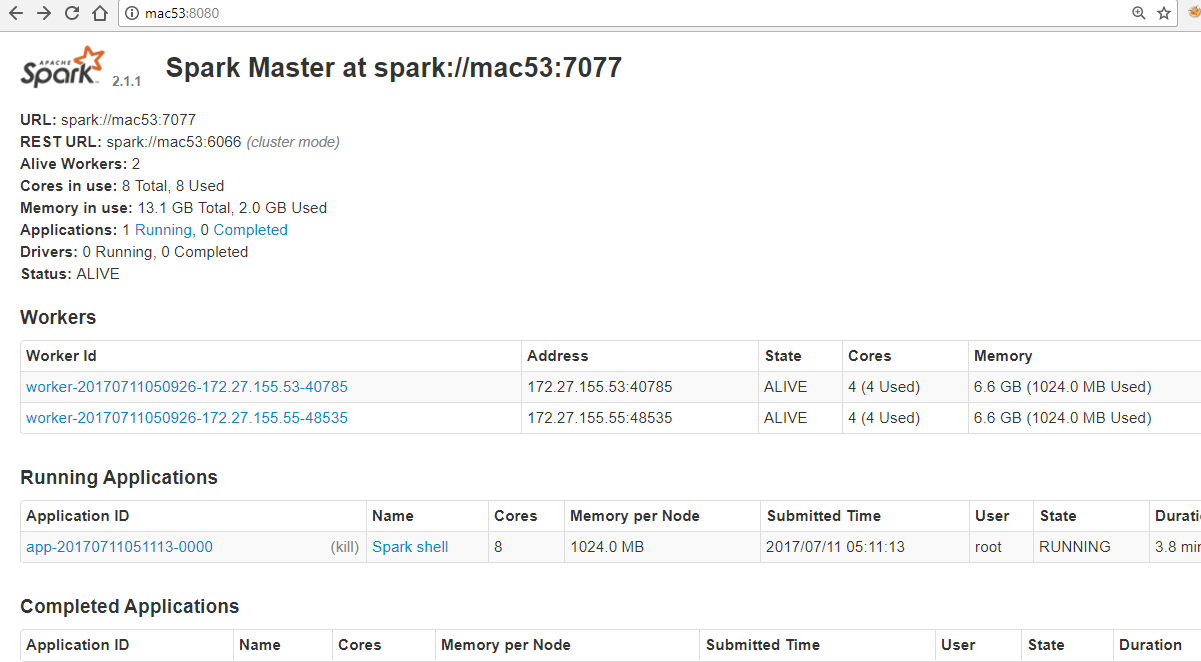
By that way we can run all worker node and master at once

Now we can run spark-shell command

|  |
| --- |
| [root@mac53 bin]# **./spark-shell --master spark://mac53:7077**  Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties  Setting default log level to "WARN".  To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  17/07/11 05:11:12 WARN SparkContext: Support for Java 7 is deprecated as of Spark 2.0.0  17/07/11 05:11:12 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable  17/07/11 05:11:14 WARN General: Plugin (Bundle) "org.datanucleus.api.jdo" is already registered. Ensure you dont have multiple JAR versions of the same plugin in the classpath. The URL "file:/home/spark/jars/datanucleus-api-jdo-3.2.6.jar" is already registered, and you are trying to register an identical plugin located at URL "file:/home/mac53/spark-2.1.1-bin-hadoop2.7/jars/datanucleus-api-jdo-3.2.6.jar."  17/07/11 05:11:14 WARN General: Plugin (Bundle) "org.datanucleus.store.rdbms" is already registered. Ensure you dont have multiple JAR versions of the same plugin in the classpath. The URL "file:/home/spark/jars/datanucleus-rdbms-3.2.9.jar" is already registered, and you are trying to register an identical plugin located at URL "file:/home/mac53/spark-2.1.1-bin-hadoop2.7/jars/datanucleus-rdbms-3.2.9.jar."  17/07/11 05:11:14 WARN General: Plugin (Bundle) "org.datanucleus" is already registered. Ensure you dont have multiple JAR versions of the same plugin in the classpath. The URL "file:/home/spark/jars/datanucleus-core-3.2.10.jar" is already registered, and you are trying to register an identical plugin located at URL "file:/home/mac53/spark-2.1.1-bin-hadoop2.7/jars/datanucleus-core-3.2.10.jar."  17/07/11 05:11:17 WARN ObjectStore: Failed to get database global\_temp, returning NoSuchObjectException  Spark context Web UI available at http://172.27.155.53:4040  Spark context available as 'sc' (master = spark://mac53:7077, app id = app-20170711051113-0000).  Spark session available as 'spark'.  Welcome to  \_\_\_\_ \_\_  / \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_  \_\ \/ \_ \/ \_ `/ \_\_/ '\_/  /\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 2.1.1  /\_/  Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.7.0\_111)  Type in expressions to have them evaluated.  Type :help for more information. |

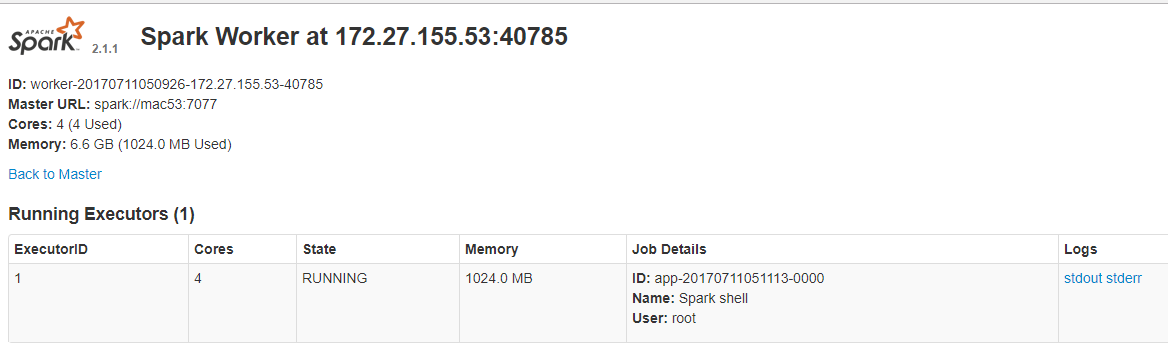
*Note: I can run only one instance of spark-shell at a time in standalone mode*

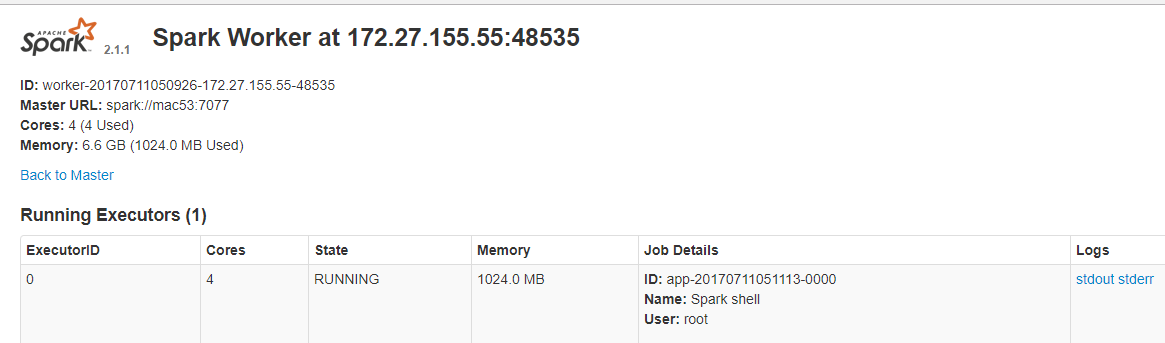
Now checks spark UI



We can see Running Application Spark-shell now

Now check the workers on spark UI





To stop the cluster, run use stop-all.sh file on your master node (mac53)

|  |
| --- |
| [root@mac53 sbin]# **./stop-all.sh**  mac53: stopping org.apache.spark.deploy.worker.Worker  mac55: stopping org.apache.spark.deploy.worker.Worker  stopping org.apache.spark.deploy.master.Master |

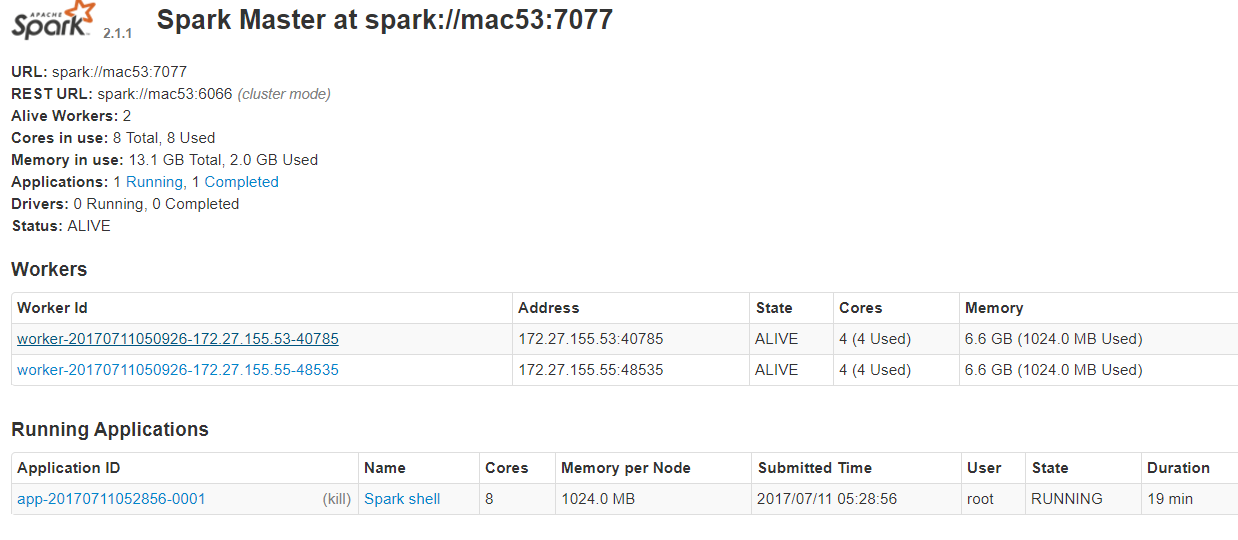
## Run job on standalone mode

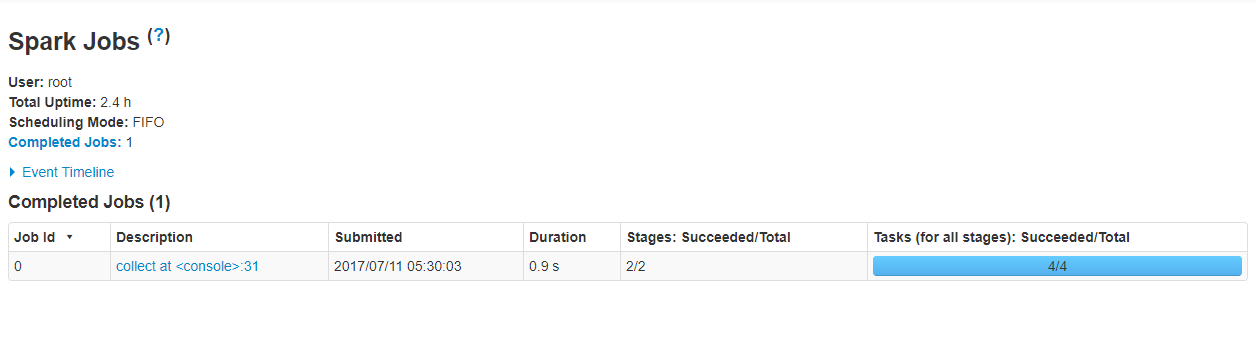
To read a file using spark-context in standalone mode we need to copy the file on same location of all worker nodes.

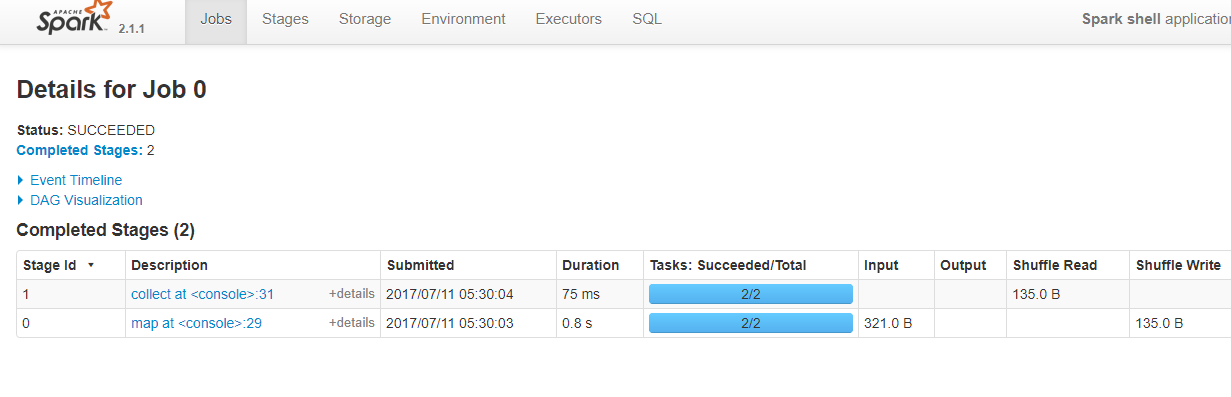
Create input.txt file at location /home/spark-input on both mac53 and mac55

|  |
| --- |
| INFO This is a message with content  INFO This is some other content  INFO Here are more messages  WARN This is a warning  ERROR Something bad happened  WARN More details on the bad thing  INFO back to normal messages |

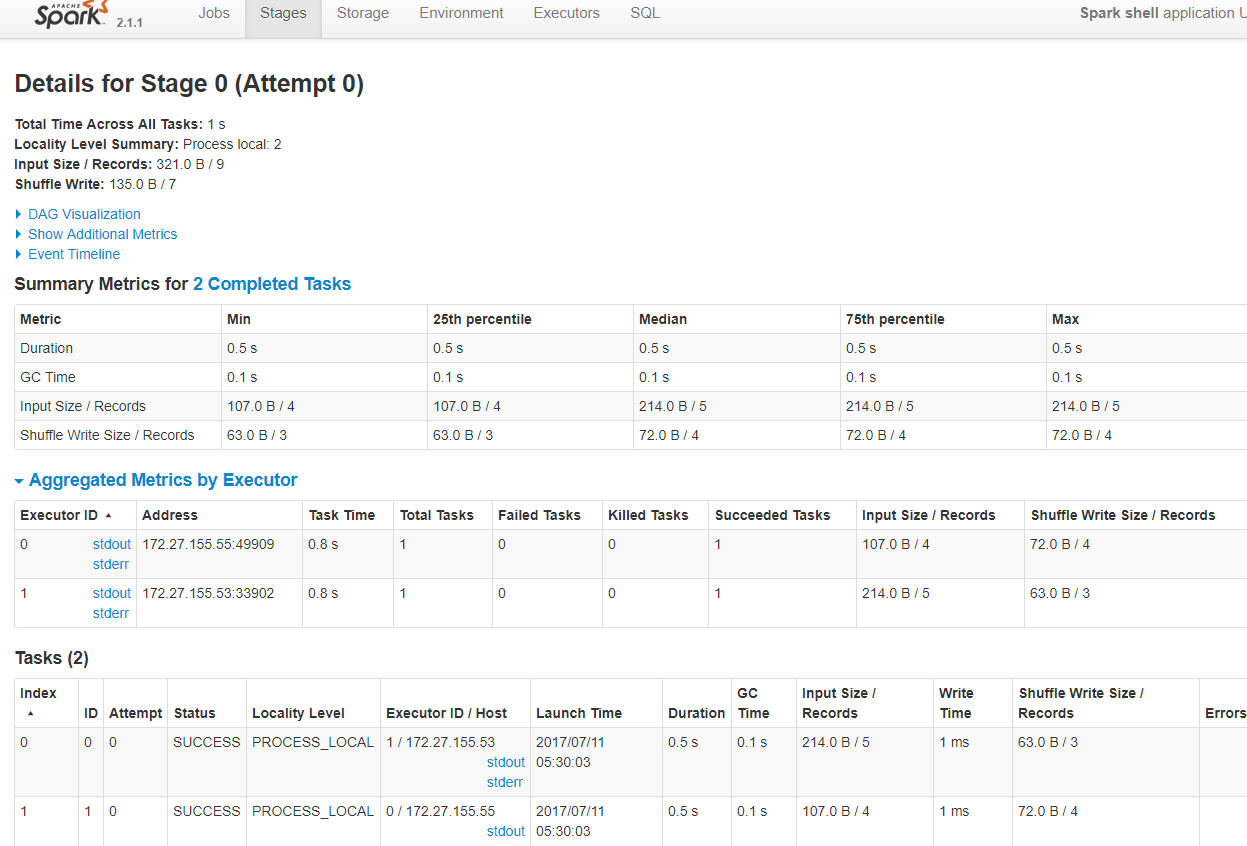
|  |
| --- |
| scala> val input = sc.textFile("/home/spark-input/input.txt")  input: org.apache.spark.rdd.RDD[String] = /home/spark-input/input.txt MapPartitionsRDD[1] at textFile at <console>:24  scala> val tokenized = input.  | map(line => line.split(" ")).  | filter(words => words.size > 0)  tokenized: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[3] at filter at <console>:28  scala> val counts = tokenized.  | map(words => (words(0), 1)).  | reduceByKey{ (a, b) => a + b }  counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[5] at reduceByKey at <console>:30  scala> counts.collect  res0: Array[(String, Int)] = Array((ERROR,1), (INFO,4), ("",2), (WARN,2)) |

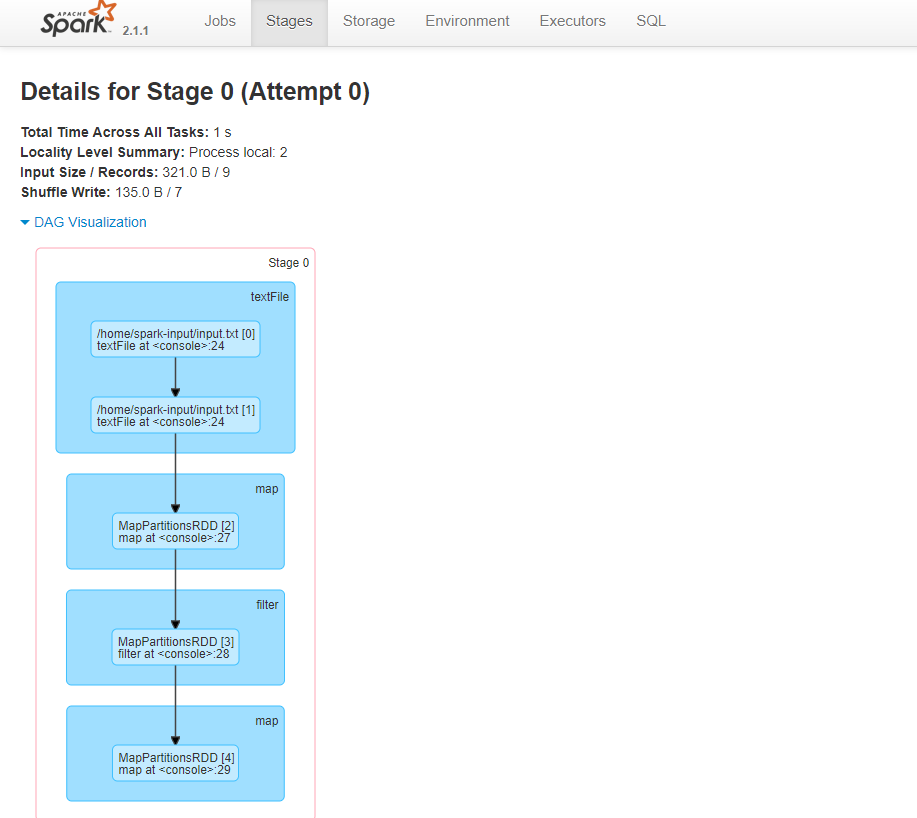




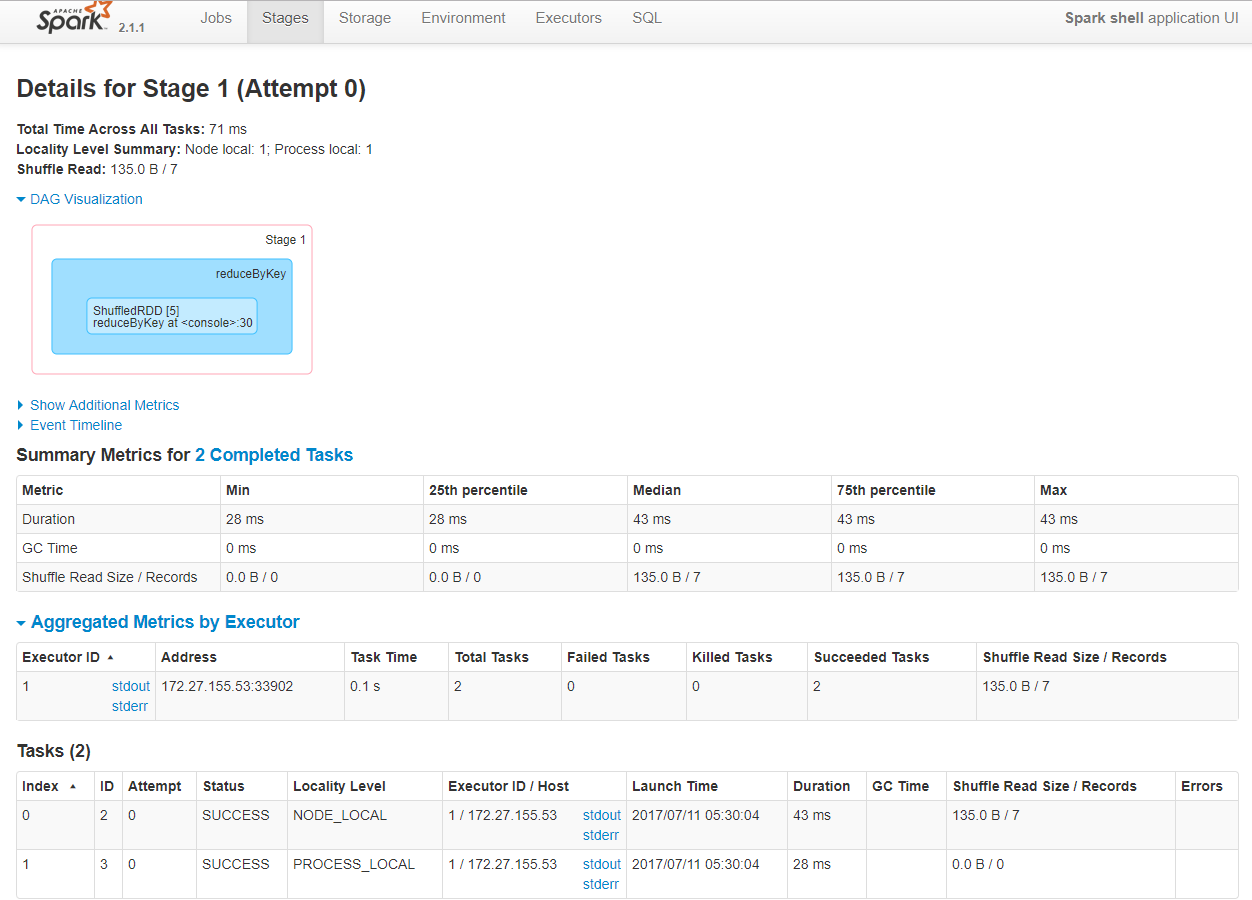


#### First Stage





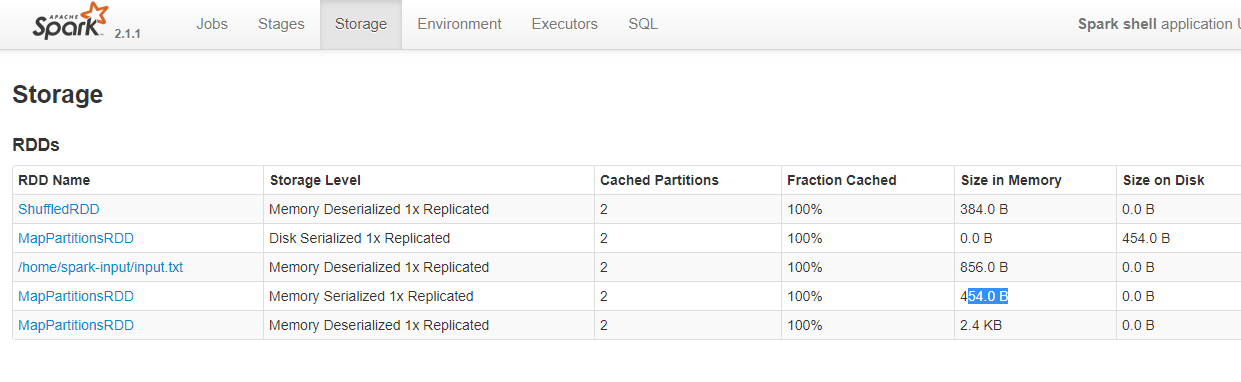
#### Second Stage



When we persist data on disk it will be stored on disk file (spark.local.dir (by default /tmp)

|  |
| --- |
| tokenized.persist(StorageLevel. DISK\_ONLY) |

Remember it will store only when we fire any action on that rdd and then only we can see it under storage tab of spark UI



## Spark-sql shell on standalone mode

First copy **hive-site.xml** file from **/etc/hive/conf** folder to **<spark-home>/conf** folder

You can run spark-sql present under **<spark-home>/bin** folder

|  |
| --- |
| [root@mac53 bin]# ./spark-sql  17/07/12 06:55:18 INFO HiveClientImpl: Warehouse location for Hive client (version 1.2.1) is /user/hive/warehouse  17/07/12 06:55:18 INFO SessionState: Created local directory: /tmp/17d88b8b-aa15-46db-b480-aaa842672413\_resources  17/07/12 06:55:18 INFO SessionState: Created HDFS directory: /tmp/hive/root/17d88b8b-aa15-46db-b480-aaa842672413  17/07/12 06:55:18 INFO SessionState: Created local directory: /tmp/root/17d88b8b-aa15-46db-b480-aaa842672413  17/07/12 06:55:18 INFO SessionState: Created HDFS directory: /tmp/hive/root/17d88b8b-aa15-46db-b480-aaa842672413/\_tmp\_space.db  17/07/12 06:55:18 INFO HiveClientImpl: Warehouse location for Hive client (version 1.2.1) is /user/hive/warehouse  spark-sql> |

Now we have connected with hive using spark sql ,here we can run any query

Refer link

<https://spark.apache.org/docs/latest/spark-standalone.html>