--mSBT-Scala Program Deployment Steps:

1.Install SBT

echo "deb https://dl.bintray.com/sbt/debian /" | sudo tee -a /etc/apt/sources.list.d/sbt.list

sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 642AC823

sudo apt-get update

sudo apt-get install sbt

* Create a Directory in your local system.
  + Mkdir /home/yeti/sparksample
  + Cd /home/yeti/sparksample
    - Mkdir project
    - Mkdir –p src/main/scala
* Create an sbt file in the root directory as mentioned below.
  + Vi sparksample.sbt
    - * name := "Spark Sample"

version := "1.0"

scalaVersion := "2.10.3"

libraryDependencies += "org.apache.spark" %% "spark-core" % "1.6.0"

* + Save sparksample.sbt file.
* Create a scala program “SparkPi.scala” under src/main/scala directory.
* In the root directory run below command to create a jar file.
  + Sbt package
* Once the jar file is ready, We can execute this jar file in local mode for testing and YARN mode for deploying on Azure cluster.
  + Run below commands depending on the mode of deployment.
  + <http://spark.apache.org/docs/latest/submitting-applications.html>
* *# Run application locally on 8 cores*
* ./bin/spark-submit **\**
* --class org.apache.spark.examples.SparkPi **\**
* --master local[8] **\**
* /path/to/examples.jar **\**
* 100
* *# Run on a Spark standalone cluster in client deploy mode*
* ./bin/spark-submit **\**
* --class org.apache.spark.examples.SparkPi **\**
* --master spark://207.184.161.138:7077 **\**
* --executor-memory 20G **\**
* --total-executor-cores 100 **\**
* /path/to/examples.jar **\**
* 1000
* *# Run on a Spark standalone cluster in cluster deploy mode with supervise*
* ./bin/spark-submit **\**
* --class org.apache.spark.examples.SparkPi **\**
* --master spark://207.184.161.138:7077 **\**
* --deploy-mode cluster **\**
* --supervise **\**
* --executor-memory 20G **\**
* --total-executor-cores 100 **\**
* /path/to/examples.jar **\**
* 1000
* *# Run on a YARN cluster*
* export HADOOP\_CONF\_DIR=XXX
* ./bin/spark-submit **\**
* --class org.apache.spark.examples.SparkPi **\**
* --master yarn **\**
* --deploy-mode cluster **\**  *# can be client for client mode*
* --executor-memory 20G **\**
* --num-executors 50 **\**
* /path/to/examples.jar **\**
* 1000
* *# Run a Python application on a Spark standalone cluster*
* ./bin/spark-submit **\**
* --master spark://207.184.161.138:7077 **\**
* examples/src/main/python/pi.py **\**
* 1000
* *# Run on a Mesos cluster in cluster deploy mode with supervise*
* ./bin/spark-submit **\**
* --class org.apache.spark.examples.SparkPi **\**
* --master mesos://207.184.161.138:7077 **\**
* --deploy-mode cluster **\**
* --supervise **\**
* --executor-memory 20G **\**
* --total-executor-cores 100 **\**
* http://path/to/examples.jar **\**
* 1000

# Master URLs

The master URL passed to Spark can be in one of the following formats:

|  |  |
| --- | --- |
| **Master URL** | **Meaning** |
| local | Run Spark locally with one worker thread (i.e. no parallelism at all). |
| local[K] | Run Spark locally with K worker threads (ideally, set this to the number of cores on your machine). |
| local[\*] | Run Spark locally with as many worker threads as logical cores on your machine. |
| spark://HOST:PORT | Connect to the given [Spark standalone cluster](http://spark.apache.org/docs/latest/spark-standalone.html) master. The port must be whichever one your master is configured to use, which is 7077 by default. |
| mesos://HOST:PORT | Connect to the given [Mesos](http://spark.apache.org/docs/latest/running-on-mesos.html) cluster. The port must be whichever one your is configured to use, which is 5050 by default. Or, for a Mesos cluster using ZooKeeper, use mesos://zk://.... To submit with --deploy-mode cluster, the HOST:PORT should be configured to connect to the [MesosClusterDispatcher](http://spark.apache.org/docs/latest/running-on-mesos.html" \l "cluster-mode). |
| yarn | Connect to a [YARN](http://spark.apache.org/docs/latest/running-on-yarn.html)cluster in client or cluster mode depending on the value of --deploy-mode. The cluster location will be found based on the HADOOP\_CONF\_DIR or YARN\_CONF\_DIR variable. |

1. Yetidm performance tuning

2. SPS tables loading

3. Delete Flags

4. Insert overwrite table command testing

5. sqoop and hive tables shell script for deletes

6. Remove unnecessary tables from daily jobs.

* 10\_ctas\_calendar\_hive\_v5.hql
* 10\_ctas\_company\_hive\_v5.hql