Training script for Bigtop workshop session

```
# 1. Pull the container
docker pull c0sin/bigtop-ignite:h26-i12-s13

# 2. Run the container
docker run -t -i -d -h 'ignite.docker' c0sin/bigtop-ignite:h26-i12-s13 /bin/bash
. /etc/profile.d/bigtop.sh
% cd /bigtop

# 3. make sure we are on branch-1.0
git branch

# 4. Repo setup
# skip to step 5. if it is already added to the container image
# Setup the repo for Bigtop 1.0 release
cd /etc/apt/sources.list.d
wget https://www.apache.org/dist/bigtop/bigtop-1.0.0/repos/trusty/bigtop.list
# if the signing key isn't set on your system, run the following
  sudo install debian-keyring
  gpg --recv-key AB66416ED0C3824F
  gpg --armor --export AB66416ED0C3824F | apt-key add -
# End of repo setup

# 5. Update the packages
sudo apt-get update

# 6. Build custom package for Ignite
# we'll be using Hadoop 2.6 and Spark 1.3 from the Bigtop v1.0
# we'll build our own packages for Ignite 1.3.3 from the master branch
vi bigtop.mk # change IGNITE_HADOOP_BASE_VERSION from 1.2.0 to 1.3.0. etc

./gradlew ignite-hadoop-apt

# 7. let's do the deployment
# make sure everything Puppet needs is set
./gradlew toolchain-puppetmodules
# Prepare deployment files (set in the docker image
/etc/puppet/hieradata/site.yaml);
# and deploy
sudo puppet apply -d --confdir=bigtop-deploy/puppet \
     --modulepath="bigtop-deploy/puppet/modules:/etc/puppet/modules" \
     bigtop-deploy/puppet/manifests/site.pp
```

```
# 8. Benchmarking

# 8.0 Preparations
# We need a special configuration to allow IgniteRDD to work for Spark cluster
cp /etc/ignite-hadoop/conf/igfs-config.xml
/etc/ignite-hadoop/conf/default-config.xml
service ignite-hadoop restart
export MR_JAR=/usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar

# 8.1 in-memory MR
# Run traditional MR vs in-memory MR
time hadoop jar $MR_JAR pi 20 20
time HADOOP_CONF_DIR=/etc/hadoop/ignite.client.conf hadoop jar $MR_JAR pi 20 20

# run IO-bound MR w/ IGFS and w/o

# 8.2 Do generation
time hadoop jar $MR_JAR teragen 100000 /user/root/tera.out.t
time HADOOP_CONFIG_DIR=/etc/hadoop/ignite.client.conf hadoop jar $MR_JAR \
     teragen 100000 /user/root/tera.out.imc

# 8.3 Do sorting
time hadoop jar $MR_JAR terasort /user/root/tera.out.t /user/root/sort.out.t
time HADOOP_CONF_DIR=/etc/hadoop/ignite.client.conf hadoop jar $MR_JAR \
     terasort /user/root/tera.out.imc /user/root/sort.out.imc

# 8.4 Validate (optional)
time hadoop jar $MR_JAR teravalidate /user/root/sort.out.imc \
     /user/root/validate.report.imc
time HADOOP_CONF_DIR=/etc/hadoop/ignite.client.conf hadoop jar $MR_JAR \
     teravalidate /user/root/sort.out.imc /user/root/validate.report.imc
```

```
# 9. Sharing the state between Spark jobs using Ignite Fabric
# Spark 1.3.1 (w/ Ignite 1.3+)

# 9.1 shutdown running Ignite service
service ignite-hadoop stop
###!!! There are some permissions issues, that haven't been fixed in Bigtop 1.0
###    But our docker image has all the right patches in place to do
##% chmod a+w /usr/lib/ignite-hadoop/work
##% mkdir -p /tmp/ignite/work && chmod a+w /tmp/ignite/work

# 9.2 Run a server node with spark-ignite configuration
sudo -u spark DEFAULT_CONFIG=/bigtop/spark-ignite-config.xml \
     /usr/lib/ignite-hadoop/bin/ignite.sh 2>&1 > /tmp/nshell.log &

# 9.3 Start Spark Shell
sudo -u spark spark-shell --packages
org.apache.ignite:ignite-spark:1.3.3-p2-SNAPSHOT \
     --repositories https://repository.apache.org/content/groups/snapshots \
     --master spark://ignite.docker:7077

# 9.4 Execute these commands in the shell
import org.apache.ignite.spark._
import org.apache.ignite.configuration._

val ic = new IgniteContext[Integer, Integer](sc, "/bigtop/spark-ignite-config.xml")

val sharedRDD = ic.fromCache("SharedNumbers")

sharedRDD.filter(_._2 < 10).collect()
sharedRDD.savePairs(sc.parallelize(1 to 100000, 10).map(i => (i, i*2)))

sharedRDD.filter(_._2 > 90000).count

sharedRDD.sql("select count(_val) from Integer where _val > ?", 90000).collect()

# 9.5 Restart (stop & start) Spark Shell with the command 9.3
# Note that commands below don't store state, only read the data from existing
Ignite cache

# 9.6 Execute these commands
import org.apache.ignite.spark._
import org.apache.ignite.configuration._

val ic = new IgniteContext[Integer, Integer](sc, "/bigtop/spark-ignite-config.xml")

val sharedRDD = ic.fromCache("SharedNumbers")
// Now it just works with the data which has been stored in the cache already
sharedRDD.filter(_._2 > 90000).count

sharedRDD.sql("select count(_val) from Integer where _val > ?", 90000).collect()

# The end
```