Hands on Exercises for Cloudera Hadoop-Developer Track

Table of Contents

Exercise1: Setting up Environment

Step1: Installing VMware Player from Training Bundle.

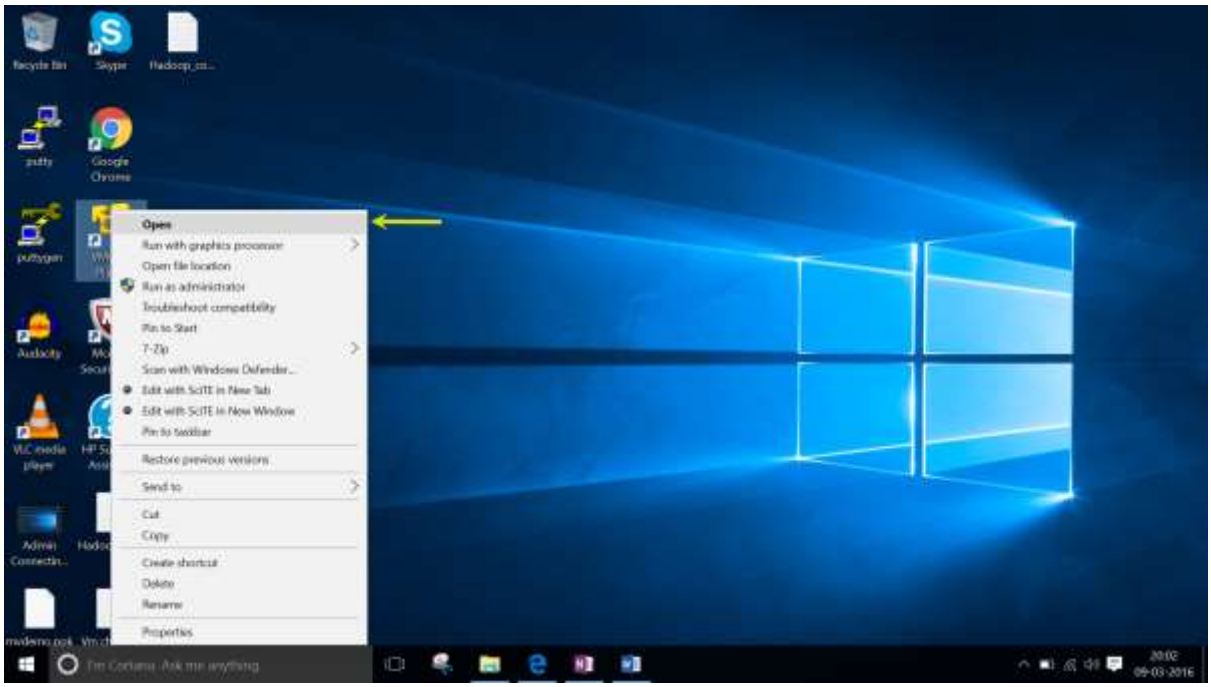
- Under 'Training_Bundle_For_Hortonworks_Developer_Track' folder, navigate to "Additional Software" folder.
- Find the executable file named 'VmWarePlayer.exe' and complete the installation.

Step2: Extracting the VMware Image

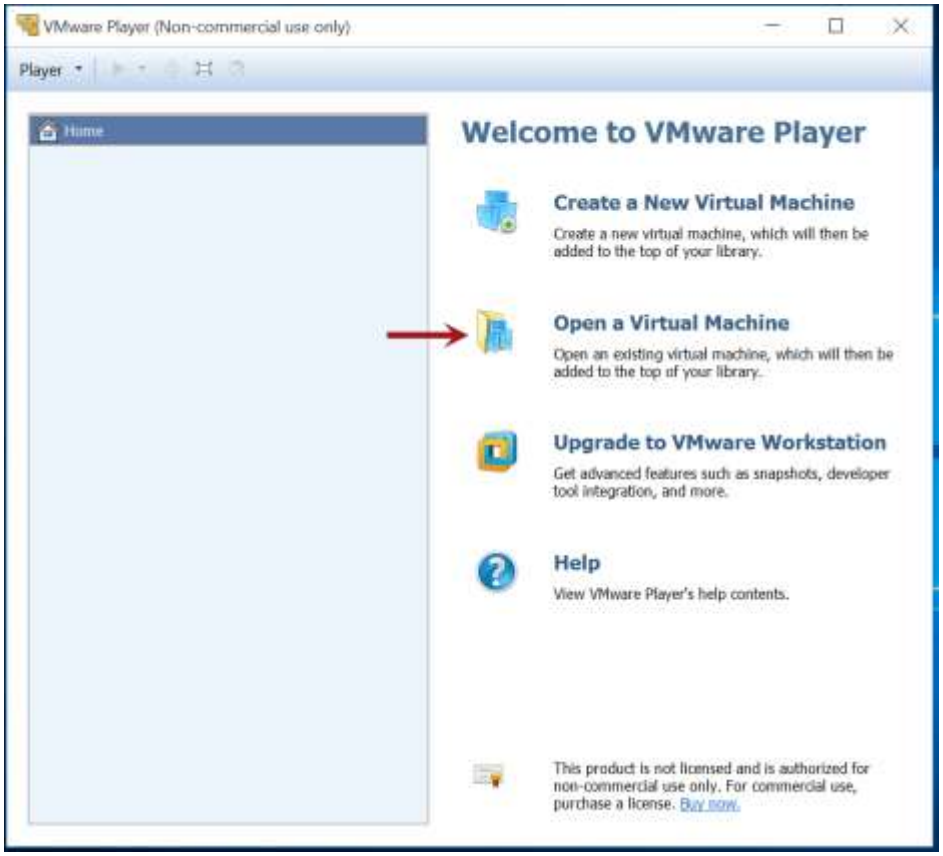
- From the same training bundle, locate the folder named VM image> 'technocrafty-quickstart-vm-5.5.0-0-vmware'
- Unzip/Extract the files and using the VMware player browse to this location to get start with virtual machine.

Step3: Loading the VMware Image

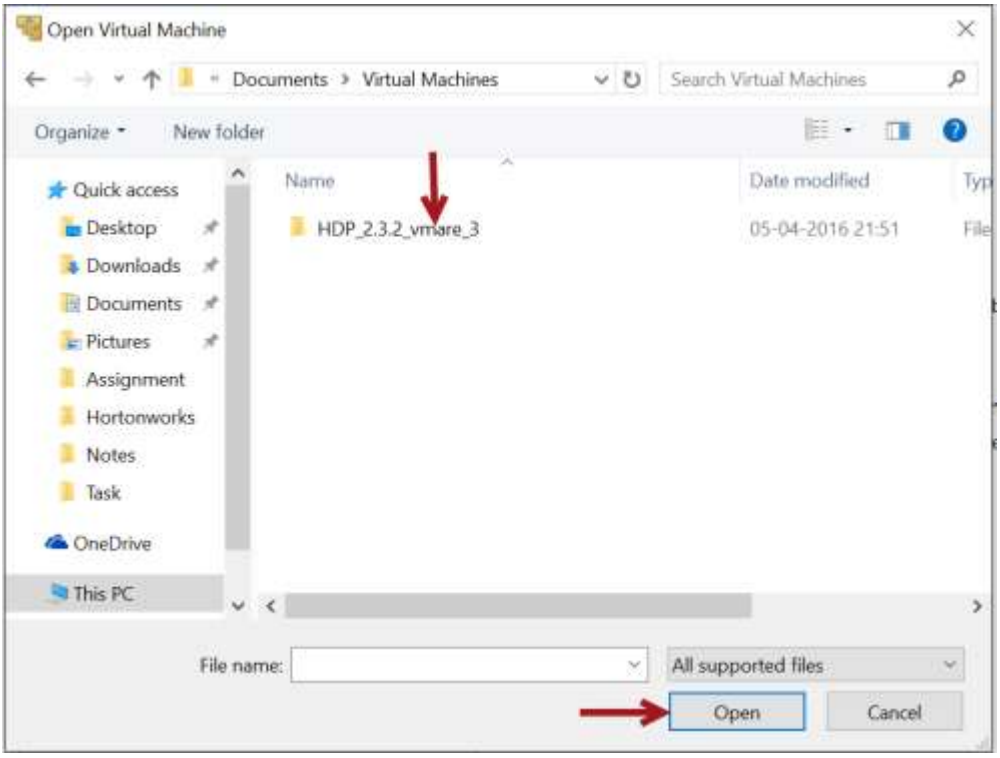
- Right click and select Open or Double click on 'VMware Player' and launch the same.

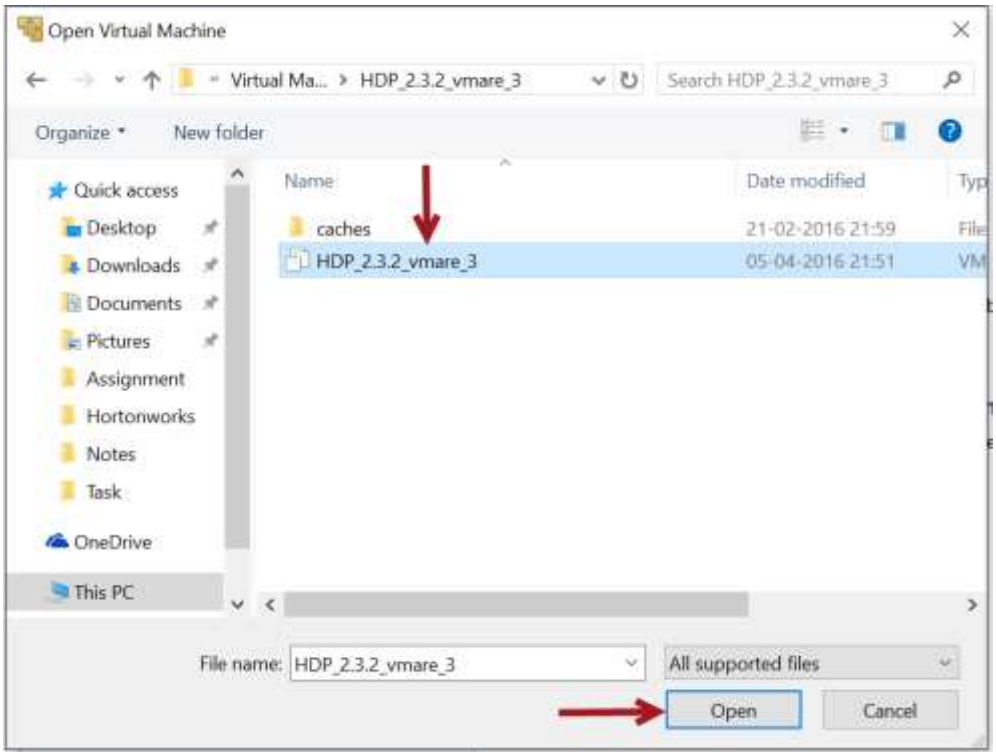


- Select the option as “Open a Virtual Machine”

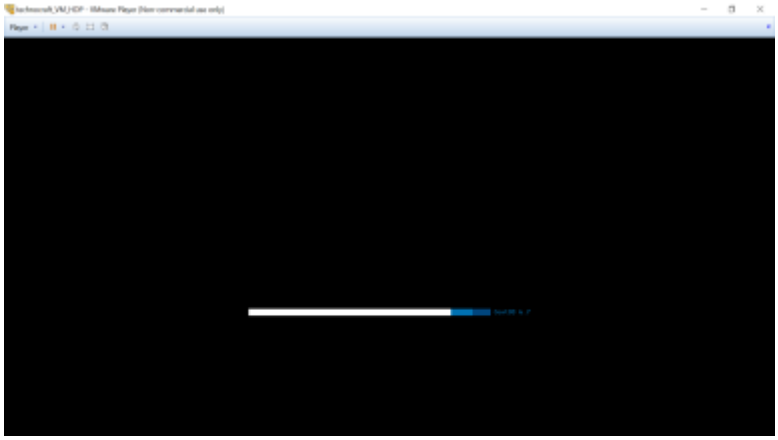


- Navigate to the location where you have extracted the VMware Image in earlier step.


Open Virtual Machine
Documents › Virtual Machines
Search Virtual Machines
Organize
New folder
Quick access
Desktop
Downloads
Documents
Pictures
Assignment
Hortonworks
Notes
Task
OneDrive
This PC
Name
Date modified
HDP_2.3.2_vmare_3
05-04-2016 21:51
File name:
All supported files
Open
Cancel


Open Virtual Machine
Virtual Ma... › HDP_2.3.2_vmare_3
Search HDP_2.3.2_vmare_3
Organize
New folder
Quick access
Desktop
Downloads
Documents
Pictures
Assignment
Hortonworks
Notes
Task
OneDrive
This PC
Name
Date modified
caches
21-02-2016 21:59
HDP_2.3.2_vmare_3
05-04-2016 21:51
File name:
HDP_2.3.2_vmare_3
All supported files
Open
Cancel

- This will start the Virtual Machine



Login with user ‘cloudera’ with password ‘cloudera’

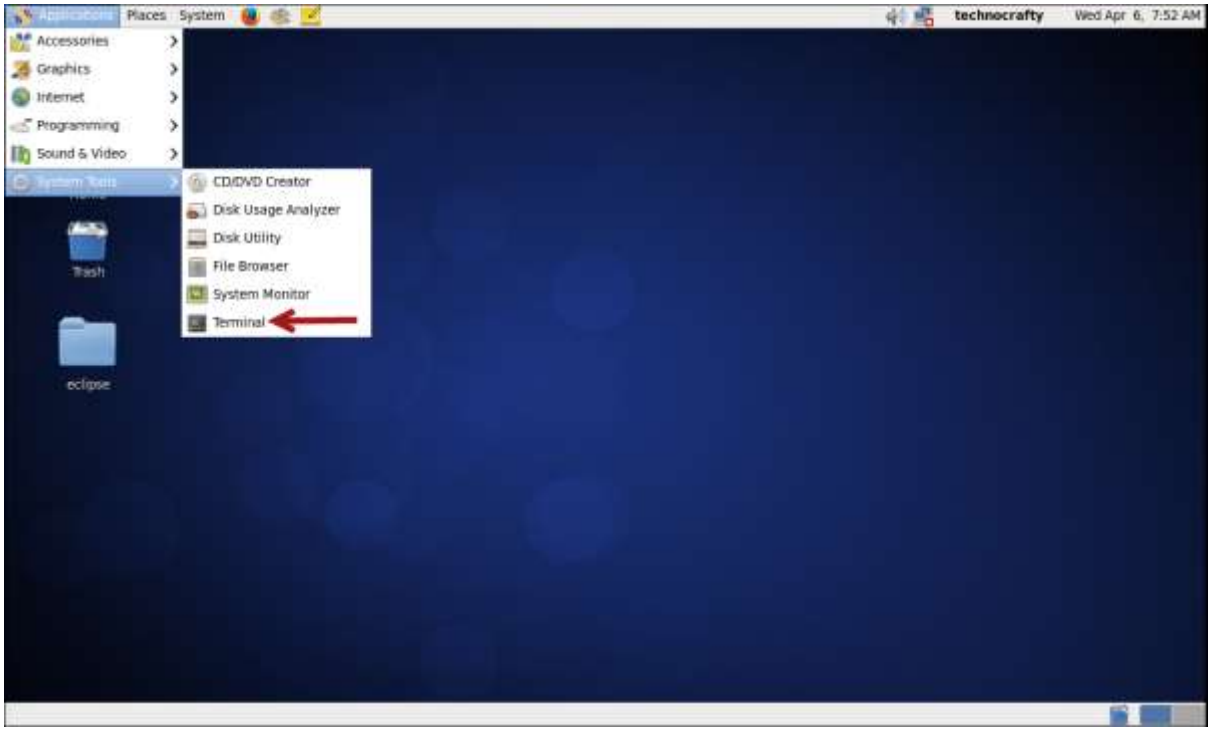
Exercise 2: Practicing HDFS Commands

We will practise HDFS command line interface and web-based Hue File Browser to perform operations on files.

HDFS Command line Interface

- Open terminal

Navigate to Application > System Tools > Terminal



Operations on Files and Directories:

To view the content of root directory in HDFS

```
[technocrafty@sandbox ~]$ hdfs dfs -ls /
Found 9 items
drwxrwxrwx   - yarn   hadoop          0 2015-10-27 14:39 /app-logs
drwxr-xr-x   - hdfs   hdfs            0 2015-10-27 15:16 /apps
drwxr-xr-x   - hdfs   hdfs            0 2015-10-27 15:04 /demo
drwxr-xr-x   - hdfs   hdfs            0 2015-10-27 14:39 /hdp
drwxr-xr-x   - mapred hdfs            0 2015-10-27 14:39 /mapred
drwxrwxrwx   - mapred hadoop          0 2015-10-27 14:40 /mr-history
drwxr-xr-x   - hdfs   hdfs            0 2015-10-27 15:09 /ranger
drwxrwxrwx   - hdfs   hdfs            0 2016-03-12 14:00 /tmp
drwxr-xr-x   - hdfs   hdfs            0 2016-03-05 08:22 /user
[technocrafty@sandbox ~]$
```

To view the content of /user directory

```
[technocrafty@sandbox ~]$ hdfs dfs -ls /user
Found 12 items
drwxrwx---   - ambari-qa    hdfs                0 2015-10-27 14:39 /user/ambari-qa
drwxr-xr-x   - guest        guest               0 2015-10-27 14:55 /user/guest
drwxr-xr-x   - hcat         hdfs                0 2015-10-27 14:43 /user/hcat
drwx------   - hdfs         hdfs                0 2015-10-27 15:18 /user/hdfs
drwx------   - hive         hdfs                0 2015-10-27 14:43 /user/hive
drwxrwxrwx   - hue          hdfs                0 2015-10-27 14:55 /user/hue
drwxrwxr-x   - oozie        hdfs                0 2015-10-27 14:44 /user/oozie
drwxr-xr-x   - solr         hdfs                0 2015-10-27 14:49 /user/solr
drwxrwxr-x   - spark        hdfs                0 2015-10-27 14:40 /user/spark
drwxr-xr-x   - technocrafty technocrafty        0 2016-03-05 08:22 /user/technocrafty
drwxr-xr-x   - unit         hdfs                0 2015-10-27 14:45 /user/unit
drwxr-xr-x   - zeppelin     zeppelin            0 2015-10-27 15:16 /user/zeppelin
[technocrafty@sandbox ~]$
```

Note: For an empty directory the prompt does not show any error while querying whereas if the directory doesn't exist it will throw an error.

Uploading Files to HDFS

- Create a directory named 'input' in HDFS, we will use this directory throughout the exercises.
- Ensure that directory is created by running ls command

```
[technocrafty@sandbox ~]$ hdfs dfs -mkdir input
[technocrafty@sandbox ~]$ hdfs dfs -ls
Found 1 items
drwxr-xr-x   - technocrafty technocrafty          0 2016-04-06 08:29 input
[technocrafty@sandbox ~]$
```

- Locate test file named 'sample.txt' under Datasets directory on local filesystem and upload into HDFS

```
[technocrafty@sandbox Datasets]$ pwd
/home/technocrafty/Datasets
[technocrafty@sandbox Datasets]$ ls
sample.txt
[technocrafty@sandbox Datasets]$ hdfs dfs -put sample.txt input
[technocrafty@sandbox Datasets]$ hdfs dfs -ls input
Found 1 items
-rw-r--r--   3 technocrafty technocrafty        511 2016-04-06 08:41 input/sample.txt
[technocrafty@sandbox Datasets]$ hdfs dfs -cat input/sample.txt | head -n 5
Apache Hadoop is an open source framework for distributed storage and processing of large sets of d
ata on commodity hardware. Hadoop enables businesses to quickly gain in
f structured and unstructured data.

Numerous Apache Software Foundation projects make up the services required by an enterprise to depl
oy, integrate and work with Hadoop.  Each project has been developed to deliver an explicit functio
n and each has its own community of developers and individual release cycles.
[technocrafty@sandbox Datasets]$
```

On Local FileSystem

To list top 5 lines of the file

- To download the file from HDFS to local filesystem

```
[technocrafty@sandbox Datasets]$ hdfs dfs -get /user/technocrafty/input/sample.txt \
> /home/technocrafty
16/04/06 08:50:24 WARN hdfs.DFSClient: DFSInputStream has been closed already
[technocrafty@sandbox Datasets]$ cd
[technocrafty@sandbox ~]$ ls
Datasets  Documents  Music     Public      Templates
Desktop   Downloads  Pictures  sample.txt  Videos     workspace
[technocrafty@sandbox ~]$ pwd
/home/technocrafty
[technocrafty@sandbox ~]$
```

\ indicates that command is continuing in next line

- To remove the file from HDFS

```
[technocrafty@sandbox ~]$ hdfs dfs -ls input/sample.txt
-rw-r--r--   3 technocrafty technocrafty        511 2016-04-06 08:41 input/sample.txt
[technocrafty@sandbox ~]$ hdfs dfs -rm -r input/sample.txt
16/04/06 09:34:03 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 360
 minutes, Emptier interval = 0 minutes.
Moved: 'hdfs://sandbox.technocrafty:8020/user/technocrafty/input/sample.txt' to trash at: hdfs://sa
ndbox.technocrafty:8020/user/technocrafty/.Trash/Current
[technocrafty@sandbox ~]$ hdfs dfs -ls input/sample.txt
ls: `input/sample.txt': No such file or directory
[technocrafty@sandbox ~]$
```

Web-Based Interface (Hue File Browser)

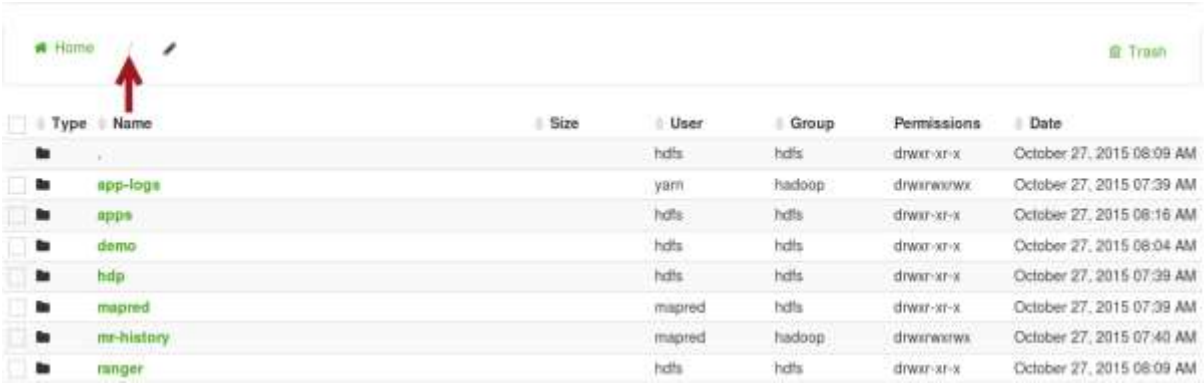
- Open a web browser on your VM, and navigate to bookmark tab and select HUE. This can be alternatively launched by specifying

- Enter username as ‘cloudera’ and password ‘cloudera’.

- Navigate to file browser

By default, the content of your HDFS home directory will be displayed. Locate the input directory created in previous step on the file browser.

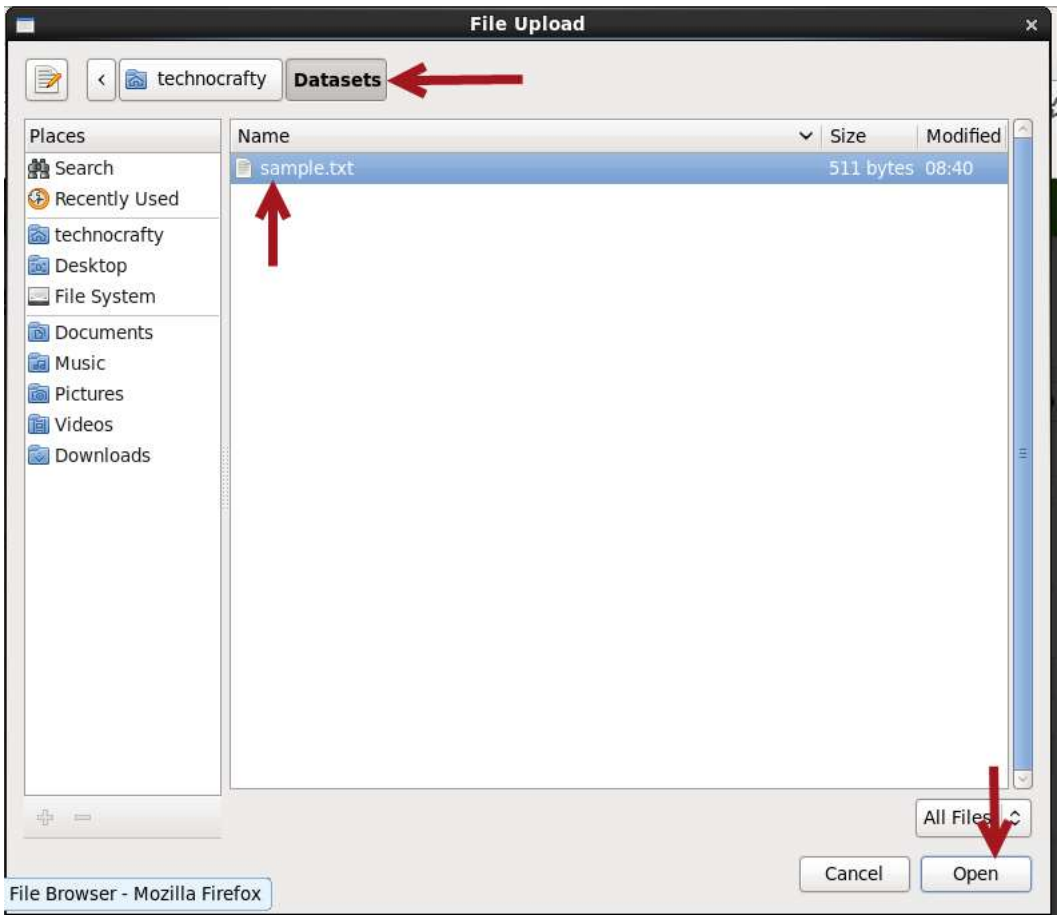
- Click on leading slash to view the content of root directory

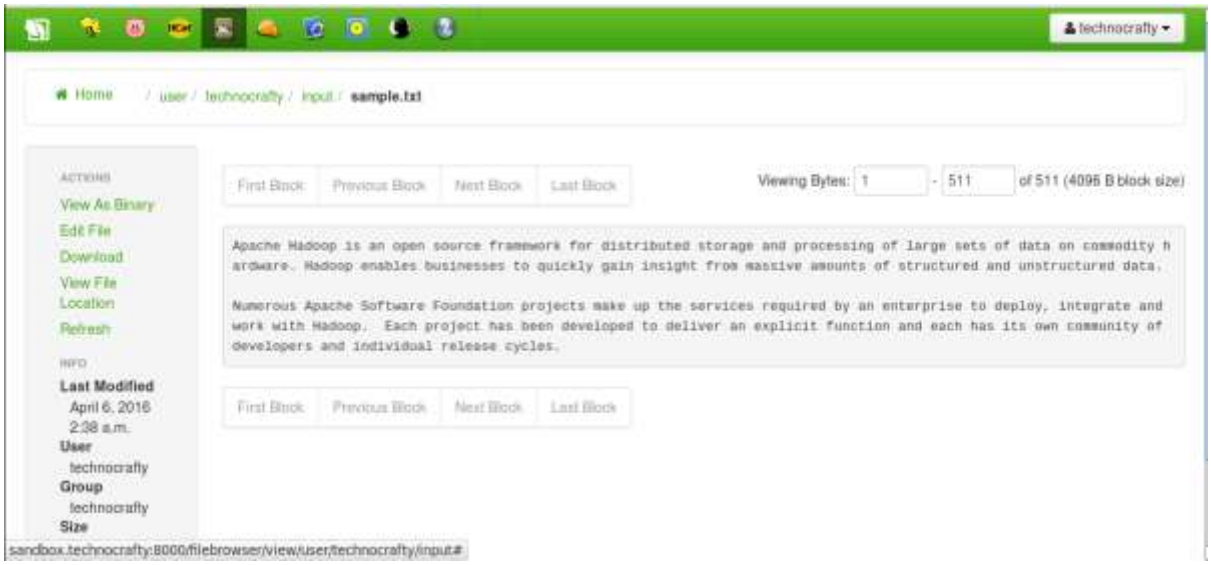


- To perform operation on a file, select the file and several options will be enabled

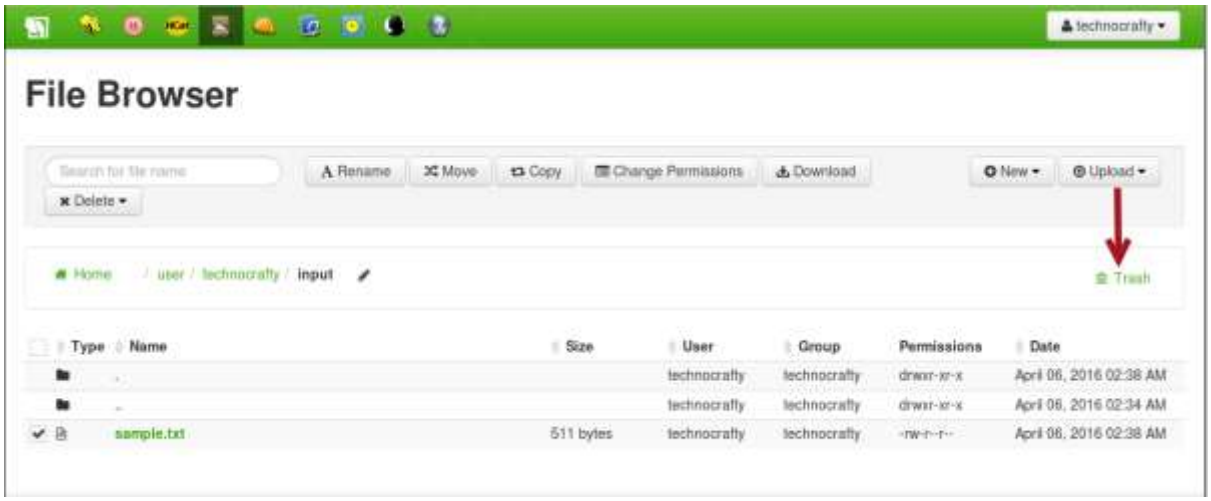
- To upload the file, click on Upload button. From the drop down you can choose to upload a plain file or zipped file

- Click on Upload > Files > Select files


File Upload
technocrafty
Datasets
Places
Search
Recently Used
technocrafty
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
sample.txt
511 bytes
08:40
All Files
Cancel
Open
File Browser - Mozilla Firefox


technocrafty
Home / user / technocrafty / input / sample.txt
View As Binary
Edit File
Download
View File Location
Refresh
Last Modified
April 6, 2016 2:38 a.m.
User
technocrafty
Group
technocrafty
Size
First Block
Previous Block
Next Block
Last Block
Viewing Bytes: 1 - 511 of 511 (4096 B block size)
Apache Hadoop is an open source framework for distributed storage and processing of large sets of data on commodity h
ardware. Hadoop enables businesses to quickly gain insight from massive amounts of structured and unstructured data.
Numerous Apache Software Foundation projects make up the services required by an enterprise to deploy, integrate and
work with Hadoop. Each project has been developed to deliver an explicit function and each has its own community of
developers and individual release cycles.
sandbox.technocrafty:8000/filebrowser/view/user/technocrafty/input#

- To remove the file, move it to Trash



Exercise 3: Spark

Overview
In this lab, we will look at several transformations on RDD
Builds on
Previous labs for the transformations we'll use.
Run time
20-30 minutes

- Launch spark-shell
- Spark shell can be launched in two ways i.e. Scala and Python.
- To launch Scala spark shell, follow below steps

```
$ cd <path>/spark-2.4.4-bin-hadoop2.6

$ bin/spark-shell
```

- Spark creates a SparkContext object called sc, verify that the object exists

```
scala> sc
res0: org.apache.spark.SparkContext = org.apache.spark.SparkContext@8e75504
```

- To know various SparkContext methods which are available, type sc. (sc followed by dot) and then TAB key

```
scala> sc.
accumulable                    accumulableCollection          accumulator
addFile                        addJar                         addSparkListener
appName                        applicationAttemptId           applicationId
asInstanceOf                   binaryFiles                    binaryRecords
broadcast                      cancelAllJobs                  cancelJobGroup
clearCallSite                  clearFiles                     clearJars
clearJobGroup                  defaultMinPartitions           defaultMinSplits
defaultParallelism             emptyRDD                       externalBlockStoreFolderName
files                          getAllPools                    getCheckpointDir
getConf                        getExecutorMemoryStatus        getExecutorStorageStatus
getLocalProperty               getPersistentRDDs              getPoolForName
getRDDStorageInfo              getSchedulingMode              hadoopConfiguration
hadoopFile                     hadoopRDD                      initLocalProperties
isInstanceOf                   isLocal                        jars
killExecutor                   killExecutors                  makeRDD
master                         metricsSystem                  newAPIHadoopFile
newAPIHadoopRDD                objectFile                     parallelize
range                          requestExecutors               runApproximateJob
runJob                         sequenceFile                   setCallSite
setCheckpointDir               setJobDescription              setJobGroup
setLocalProperty               setLogLevel                    sparkUser
startTime                      statusTracker                  stop
submitJob                      tachyonFolderName              textFile
toString                       union                          version
wholeTextFiles
```

1. **Basic Commands and Operations**

Spark is based on the concept of Resilient Distributed Dataset (RDD), which is fault tolerant collection of elements that can be operated in parallel.

Two ways to create RDDs:

- Parallelizing an existing collection
- Referencing a dataset from an External Storage System

Parallelized collection:

To create a parallelized collection holding numbers 1 to 6

Example 1

val data = Array (1,2,3,4,5,6)

val distData = sc.parallelize(data)

```
scala> val data = Array (1,2,3,4,5,6)
data: Array[Int] = Array(1, 2, 3, 4, 5, 6)

scala> val distData = sc.parallelize(data)
distData: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:17
```

Once 'distData' dataset is created, we can perform operations such as:

- Finding the sum, mean & variance

```
distData.sum

distData.mean

distData.variance
```

```
scala> distData.sum
16/04/16 15:08:23 INFO SparkContext: Starting job: sum at <console>:20
16/04/16 15:08:23 INFO DAGScheduler: Got job 1 (sum at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 15:08:23 INFO DAGScheduler: Final stage: ResultStage 1(sum at <console>:20)
16/04/16 15:08:23 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:08:23 INFO DAGScheduler: Missing parents: List()
16/04/16 15:08:23 INFO DAGScheduler: Submitting ResultStage 1 (MapPartitionsRDD[2] at numericRDDToDoubleRDDFunctions at <console
rents
16/04/16 15:08:23 INFO MemoryStore: ensureFreeSpace(2336) called with curMem=3788, maxMem=278302556
16/04/16 15:08:23 INFO MemoryStore: Block broadcast_1 stored as values in memory (estimated size 2.3 KB, free 265.4 MB)
16/04/16 15:08:23 INFO MemoryStore: ensureFreeSpace(1452) called with curMem=6124, maxMem=278302556
16/04/16 15:08:23 INFO MemoryStore: Block broadcast_1_piece0 stored as bytes in memory (estimated size 1452.0 B, free 265.4 MB)
16/04/16 15:08:23 INFO BlockManagerInfo: Added broadcast_1_piece0 in memory on localhost:58083 (size: 1452.0 B, free: 265.4 MB)
16/04/16 15:08:23 INFO SparkContext: Created broadcast 1 from broadcast at DAGScheduler.scala:874
16/04/16 15:08:23 INFO DAGScheduler: Submitting 4 missing tasks from ResultStage 1 (MapPartitionsRDD[2] at numericRDDToDoubleRDD
16/04/16 15:08:23 INFO TaskSchedulerImpl: Adding task set 1.0 with 4 tasks
16/04/16 15:08:23 INFO TaskSetManager: Starting task 0.0 in stage 1.0 (TID 4, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:08:23 INFO TaskSetManager: Starting task 1.0 in stage 1.0 (TID 5, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:08:23 INFO TaskSetManager: Starting task 2.0 in stage 1.0 (TID 6, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:08:23 INFO TaskSetManager: Starting task 3.0 in stage 1.0 (TID 7, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:08:23 INFO Executor: Running task 0.0 in stage 1.0 (TID 4)
16/04/16 15:08:23 INFO Executor: Finished task 0.0 in stage 1.0 (TID 4). 668 bytes result sent to driver
16/04/16 15:08:23 INFO Executor: Running task 1.0 in stage 1.0 (TID 5)
16/04/16 15:08:23 INFO Executor: Running task 2.0 in stage 1.0 (TID 6)
16/04/16 15:08:23 INFO Executor: Running task 3.0 in stage 1.0 (TID 7)
16/04/16 15:08:23 INFO TaskSetManager: Finished task 0.0 in stage 1.0 (TID 4) in 24 ms on localhost (1/4)
16/04/16 15:08:23 INFO Executor: Finished task 3.0 in stage 1.0 (TID 7). 668 bytes result sent to driver
16/04/16 15:08:23 INFO Executor: Finished task 1.0 in stage 1.0 (TID 5). 668 bytes result sent to driver
16/04/16 15:08:23 INFO Executor: Finished task 2.0 in stage 1.0 (TID 6). 668 bytes result sent to driver
16/04/16 15:08:23 INFO TaskSetManager: Finished task 3.0 in stage 1.0 (TID 7) in 38 ms on localhost (2/4)
16/04/16 15:08:23 INFO TaskSetManager: Finished task 1.0 in stage 1.0 (TID 5) in 33 ms on localhost (3/4)
16/04/16 15:08:23 INFO TaskSetManager: Finished task 2.0 in stage 1.0 (TID 6) in 33 ms on localhost (4/4)
16/04/16 15:08:23 INFO TaskSchedulerImpl: Removed TaskSet 1.0, whose tasks have all completed, from pool
16/04/16 15:08:23 INFO DAGScheduler: ResultStage 1 (sum at <console>:20) finished in 0.085 s
16/04/16 15:08:23 INFO DAGScheduler: Job 1 finished: sum at <console>:20, took 0.053322 s
res2: Double = 21.0
```

```
scala> distData.mean
16/04/16 15:09:57 INFO SparkContext: Starting job: mean at <console>:20
16/04/16 15:09:57 INFO DAGScheduler: Got job 2 (mean at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 15:09:57 INFO DAGScheduler: Final stage: ResultStage 2(mean at <console>:20)
16/04/16 15:09:57 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:09:57 INFO DAGScheduler: Missing parents: List()
16/04/16 15:09:57 INFO DAGScheduler: Submitting ResultStage 2 (MapPartitionsRDD[4] at mean at <console>:20), which has no missing parents
16/04/16 15:09:57 INFO MemoryStore: ensureFreeSpace(2496) called with curMem=7576, maxMem=278302556
16/04/16 15:09:57 INFO MemoryStore: Block broadcast_2 stored as values in memory (estimated size 2.4 KB, free 265.4 MB)
16/04/16 15:09:57 INFO MemoryStore: ensureFreeSpace(1526) called with curMem=10072, maxMem=278302556
16/04/16 15:09:57 INFO MemoryStore: Block broadcast_2_piece0 stored as bytes in memory (estimated size 1526.0 B, free 265.4 MB)
16/04/16 15:09:57 INFO BlockManagerInfo: Added broadcast_2_piece0 in memory on localhost:58083 (size: 1526.0 B, free: 265.4 MB)
16/04/16 15:09:57 INFO SparkContext: Created broadcast 2 from broadcast at DAGScheduler.scala:874
16/04/16 15:09:57 INFO DAGScheduler: Submitting 4 missing tasks from ResultStage 2 (MapPartitionsRDD[4] at mean at <console>:20)
16/04/16 15:09:57 INFO TaskSchedulerImpl: Adding task set 2.0 with 4 tasks
16/04/16 15:09:57 INFO TaskSetManager: Starting task 0.0 in stage 2.0 (TID 8, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:09:57 INFO TaskSetManager: Starting task 1.0 in stage 2.0 (TID 9, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:09:57 INFO TaskSetManager: Starting task 2.0 in stage 2.0 (TID 10, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:09:57 INFO TaskSetManager: Starting task 3.0 in stage 2.0 (TID 11, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:09:57 INFO Executor: Running task 1.0 in stage 2.0 (TID 9)
16/04/16 15:09:57 INFO Executor: Running task 3.0 in stage 2.0 (TID 11)
16/04/16 15:09:57 INFO Executor: Running task 0.0 in stage 2.0 (TID 8)
16/04/16 15:09:57 INFO Executor: Running task 2.0 in stage 2.0 (TID 10)
16/04/16 15:09:57 INFO Executor: Finished task 3.0 in stage 2.0 (TID 11). 785 bytes result sent to driver
16/04/16 15:09:57 INFO Executor: Finished task 0.0 in stage 2.0 (TID 8). 785 bytes result sent to driver
16/04/16 15:09:57 INFO Executor: Finished task 1.0 in stage 2.0 (TID 9). 785 bytes result sent to driver
16/04/16 15:09:57 INFO TaskSetManager: Finished task 3.0 in stage 2.0 (TID 11) in 9 ms on localhost (1/4)
16/04/16 15:09:57 INFO TaskSetManager: Finished task 0.0 in stage 2.0 (TID 8) in 11 ms on localhost (2/4)
16/04/16 15:09:57 INFO TaskSetManager: Finished task 1.0 in stage 2.0 (TID 9) in 11 ms on localhost (3/4)
16/04/16 15:09:57 INFO Executor: Finished task 2.0 in stage 2.0 (TID 10). 785 bytes result sent to driver
16/04/16 15:09:57 INFO TaskSetManager: Finished task 2.0 in stage 2.0 (TID 10) in 14 ms on localhost (4/4)
16/04/16 15:09:57 INFO DAGScheduler: ResultStage 2 (mean at <console>:20) finished in 0.013 s
16/04/16 15:09:57 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
16/04/16 15:09:57 INFO DAGScheduler: Job 2 finished: mean at <console>:20, took 0.027121 s
res3: Double = 3.5
```

```
scala> distData.variance
16/04/16 15:11:01 INFO SparkContext: Starting job: variance at <console>:20
16/04/16 15:11:01 INFO DAGScheduler: Got job 3 (variance at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 15:11:01 INFO DAGScheduler: Final stage: ResultStage 3(variance at <console>:20)
16/04/16 15:11:01 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:11:01 INFO DAGScheduler: Missing parents: List()
16/04/16 15:11:01 INFO DAGScheduler: Submitting ResultStage 3 (MapPartitionsRDD[6] at variance at <console>:20), which has no m
16/04/16 15:11:01 INFO MemoryStore: ensureFreeSpace(2496) called with curMem=11598, maxMem=278302556
16/04/16 15:11:01 INFO MemoryStore: Block broadcast_3 stored as values in memory (estimated size 2.4 KB, free 265.4 MB)
16/04/16 15:11:01 INFO MemoryStore: ensureFreeSpace(1529) called with curMem=14094, maxMem=278302556
16/04/16 15:11:01 INFO MemoryStore: Block broadcast_3_piece0 stored as bytes in memory (estimated size 1529.0 B, free 265.4 MB)
16/04/16 15:11:01 INFO BlockManagerInfo: Added broadcast_3_piece0 in memory on localhost:58083 (size: 1529.0 B, free: 265.4 MB)
16/04/16 15:11:01 INFO SparkContext: Created broadcast 3 from broadcast at DAGScheduler.scala:874
16/04/16 15:11:01 INFO DAGScheduler: Submitting 4 missing tasks from ResultStage 3 (MapPartitionsRDD[6] at variance at <console
16/04/16 15:11:01 INFO TaskSchedulerImpl: Adding task set 3.0 with 4 tasks
16/04/16 15:11:01 INFO TaskSetManager: Starting task 0.0 in stage 3.0 (TID 12, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:11:01 INFO TaskSetManager: Starting task 1.0 in stage 3.0 (TID 13, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:11:01 INFO TaskSetManager: Starting task 2.0 in stage 3.0 (TID 14, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:11:01 INFO TaskSetManager: Starting task 3.0 in stage 3.0 (TID 15, localhost, PROCESS_LOCAL, 1317 bytes)
16/04/16 15:11:01 INFO Executor: Running task 0.0 in stage 3.0 (TID 12)
16/04/16 15:11:01 INFO Executor: Running task 1.0 in stage 3.0 (TID 13)
16/04/16 15:11:01 INFO Executor: Running task 2.0 in stage 3.0 (TID 14)
16/04/16 15:11:01 INFO Executor: Running task 3.0 in stage 3.0 (TID 15)
16/04/16 15:11:01 INFO Executor: Finished task 2.0 in stage 3.0 (TID 14). 785 bytes result sent to driver
16/04/16 15:11:01 INFO Executor: Finished task 0.0 in stage 3.0 (TID 12). 785 bytes result sent to driver
16/04/16 15:11:01 INFO TaskSetManager: Finished task 2.0 in stage 3.0 (TID 14) in 10 ms on localhost (1/4)
16/04/16 15:11:01 INFO TaskSetManager: Finished task 0.0 in stage 3.0 (TID 12) in 12 ms on localhost (2/4)
16/04/16 15:11:01 INFO Executor: Finished task 1.0 in stage 3.0 (TID 13). 785 bytes result sent to driver
16/04/16 15:11:01 INFO TaskSetManager: Finished task 1.0 in stage 3.0 (TID 13) in 13 ms on localhost (3/4)
16/04/16 15:11:01 INFO Executor: Finished task 3.0 in stage 3.0 (TID 15). 785 bytes result sent to driver
16/04/16 15:11:01 INFO TaskSetManager: Finished task 3.0 in stage 3.0 (TID 15) in 18 ms on localhost (4/4)
16/04/16 15:11:01 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all completed, from pool
16/04/16 15:11:01 INFO DAGScheduler: ResultStage 3 (variance at <console>:20) finished in 0.021 s
16/04/16 15:11:01 INFO DAGScheduler: Job 3 finished: variance at <console>:20, took 0.036228 s
res4: Double = 2.9166666666666665
```

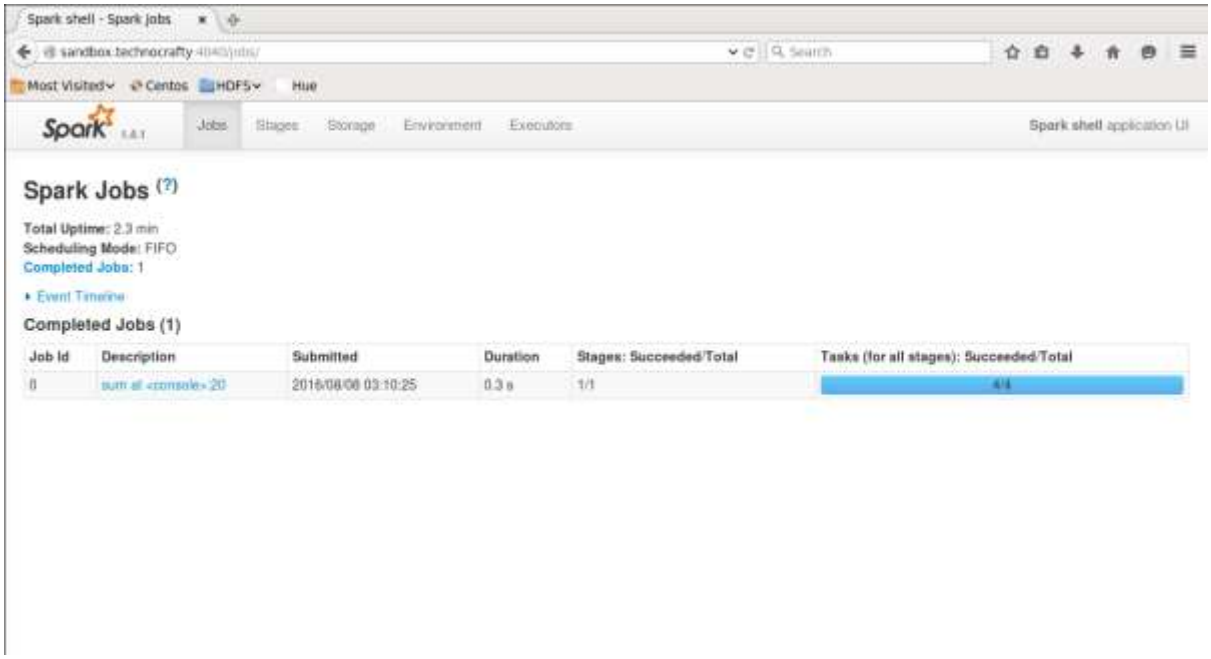
Note

Spark sets the number of partition (i.e.to cut the dataset into) automatically based on your cluster, but this can be set manually by passing second parameter to *parallelize* (e.g. sc.parallelize(data,10))

Spark UI

Spark UI can be accessed on port <localhost:4040>

Job is created once you perform an action.



External Datasets:

Spark can create distributed datasets from any storage system supported by Hadoop, Spark supports text files, SequenceFiles and any other Hadoop InputFormat.

To load a file from your Local Filesystem

Example2

```
val textFile = sc.textFile("<path>/Datasets/Employee.txt")
```

While using local filesystem, the file must be accessible at the same path on worker nodes.

Operations on RDD will be discussed in next section.

Spark's file-based input methods supports directories, compressed files and wildcards as well. i.e. textFile("/my/directory"), textFile("/my/directory/*.txt"), and textFile("/my/directory/*.gz")

2. Operations on RDD

RDDs supports two types of operations:

- **Transformations**

Creates a new dataset from an existing one

- **Actions**

Returns a value to the driver program after running a computation on the dataset.

Example: map is a transformation that passes each dataset element through a function and return a new RDD, whereas reduce is an action that aggregates all the elements of RDD using function and returns final result.

All transformations in Spark are lazy i.e. transformations are only computed when an action requires a result to be returned.

textFile dataset was already created in previous step, lets perform below operations

- counting the occurrence of lines having a particular word in it

Example: filter using scala shell

```
val textFile = sc.textFile("<path>/Datasets/Employee.txt")
textFile.count() //number of items in this RDD
textFile.first() //First  item in this RDD
val linesWithHadoop = textFile.filter(line => line.contains("7839"))
linesWithHadoop.count()
```

- Add up the sizes of all the lines

```
val lineLength = textFile.map(s => s.length)

val totalLength = lineLength.reduce((a, b) => a + b)
```

- Line with maximum words

```
Example:

textFile.map(line => line.split(" ").size).reduce((a,b) => if(a>b)   a else b)
```

```
Example4: filter

val sampleData = 1 to 5000
val totData =  sc.parallelize(sampleData)
val result = totData.filter(_ %2==0)
result.collect()


with number of partition = 2

val totDataPar = sc.parallelize(sampleData,2)
val resultPar  = totDataPar.filter(_%2==0)
resultPar.collect()
```

```
scala> val sampleData = 1 to 5000
sampleData: scala.collection.immutable.Range.Inclusive = Range(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66
, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104,
 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 13
6, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167,
168, 16...
```

```
scala> val totData =sc.parallelize(sampleData)
totData: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[12] at parallelize at <console>:17

scala> val result = totData.filter(_ %2==0)
16/04/16 15:41:40 INFO BlockManagerInfo: Removed broadcast_9_piece0 on localhost:58083 in memory (size: 1983.0 B, free: 265.4 MB)
result: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[13] at filter at <console>:19
```

```
scala> result.collect()
16/04/16 15:42:42 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 15:42:42 INFO DAGScheduler: Got job 9 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 15:42:42 INFO DAGScheduler: Final stage: ResultStage 9(collect at <console>:22)
16/04/16 15:42:42 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:42:42 INFO DAGScheduler: Missing parents: List()
16/04/16 15:42:42 INFO DAGScheduler: Submitting ResultStage 9 (MapPartitionsRDD[13] at filter at <console>:19), which has no missing parents
16/04/16 15:42:42 INFO MemoryStore: ensureFreeSpace(1944) called with curMem=210412, maxMem=278302556
16/04/16 15:42:42 INFO MemoryStore: Block broadcast_10 stored as values in memory (estimated size 1944.0 B, free 265.2 MB)
16/04/16 15:42:42 INFO MemoryStore: ensureFreeSpace(1207) called with curMem=212356, maxMem=278302556
16/04/16 15:42:42 INFO MemoryStore: Block broadcast_10_piece0 stored as bytes in memory (estimated size 1207.0 B, free 265.2 MB)
16/04/16 15:42:42 INFO BlockManagerInfo: Added broadcast_10_piece0 in memory on localhost:58083 (size: 1207.0 B, free: 265.4 MB)
16/04/16 15:42:42 INFO SparkContext: Created broadcast 10 from broadcast at DAGScheduler.scala:874
16/04/16 15:42:42 INFO DAGScheduler: Submitting 4 missing tasks from ResultStage 9 (MapPartitionsRDD[13] at filter at <console>:19)
16/04/16 15:42:42 INFO TaskSchedulerImpl: Adding task set 9.0 with 4 tasks
16/04/16 15:42:42 INFO TaskSetManager: Starting task 0.0 in stage 9.0 (TID 25, localhost, PROCESS_LOCAL, 1369 bytes)
16/04/16 15:42:42 INFO TaskSetManager: Starting task 1.0 in stage 9.0 (TID 26, localhost, PROCESS_LOCAL, 1369 bytes)
16/04/16 15:42:42 INFO TaskSetManager: Starting task 2.0 in stage 9.0 (TID 27, localhost, PROCESS_LOCAL, 1369 bytes)
16/04/16 15:42:42 INFO TaskSetManager: Starting task 3.0 in stage 9.0 (TID 28, localhost, PROCESS_LOCAL, 1426 bytes)
16/04/16 15:42:42 INFO Executor: Running task 0.0 in stage 9.0 (TID 25)
16/04/16 15:42:42 INFO Executor: Running task 1.0 in stage 9.0 (TID 26)
16/04/16 15:42:42 INFO Executor: Running task 2.0 in stage 9.0 (TID 27)
16/04/16 15:42:42 INFO Executor: Running task 3.0 in stage 9.0 (TID 28)
16/04/16 15:42:42 INFO Executor: Finished task 1.0 in stage 9.0 (TID 26). 3116 bytes result sent to driver
16/04/16 15:42:42 INFO TaskSetManager: Finished task 1.0 in stage 9.0 (TID 26) in 17 ms on localhost (1/4)
16/04/16 15:42:42 INFO Executor: Finished task 0.0 in stage 9.0 (TID 25). 3116 bytes result sent to driver
16/04/16 15:42:42 INFO TaskSetManager: Finished task 0.0 in stage 9.0 (TID 25) in 26 ms on localhost (2/4)
16/04/16 15:42:42 INFO Executor: Finished task 2.0 in stage 9.0 (TID 27). 3116 bytes result sent to driver
16/04/16 15:42:42 INFO TaskSetManager: Finished task 2.0 in stage 9.0 (TID 27) in 21 ms on localhost (3/4)
16/04/16 15:42:42 INFO Executor: Finished task 3.0 in stage 9.0 (TID 28). 3116 bytes result sent to driver
16/04/16 15:42:42 INFO TaskSetManager: Finished task 3.0 in stage 9.0 (TID 28) in 22 ms on localhost (4/4)
16/04/16 15:42:42 INFO DAGScheduler: ResultStage 9 (collect at <console>:22) finished in 0.031 s
16/04/16 15:42:42 INFO TaskSchedulerImpl: Removed TaskSet 9.0, whose tasks have all completed, from pool
16/04/16 15:42:42 INFO DAGScheduler: Job 9 finished: collect at <console>:22, took 0.041194 s
res9: Array[Int] = Array(2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 7
0, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128, 130, 132, 134, 136, 138,
```

```
scala> val totDataPar = sc.parallelize(sampleData,2)
totDataPar: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[14] at parallelize at <console>:17

scala> val resultPar= totDataPar.filter(_%2==0)
resultPar: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[15] at filter at <console>:19
```

With two partitions

```
scala> resultPar.collect()
16/04/16 15:46:15 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 15:46:15 INFO DAGScheduler: Got job 10 (collect at <console>:22) with 2 output partitions (allowLocal=false)
16/04/16 15:46:15 INFO DAGScheduler: Final stage: ResultStage 10(collect at <console>:22)
16/04/16 15:46:15 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:46:15 INFO DAGScheduler: Missing parents: List()
16/04/16 15:46:15 INFO DAGScheduler: Submitting ResultStage 10 (MapPartitionsRDD[15] at filter at <console>:19), which has no missing parents
16/04/16 15:46:15 INFO MemoryStore: ensureFreeSpace(1944) called with curMem=221563, maxMem=278302556
16/04/16 15:46:15 INFO MemoryStore: Block broadcast_11 stored as values in memory (estimated size 1944.0 B, free 265.2 MB)
16/04/16 15:46:15 INFO MemoryStore: ensureFreeSpace(1209) called with curMem=223507, maxMem=278302556
16/04/16 15:46:15 INFO MemoryStore: Block broadcast_11_piece0 stored as bytes in memory (estimated size 1209.0 B, free 265.2 MB)
16/04/16 15:46:15 INFO BlockManagerInfo: Added broadcast_11_piece0 in memory on localhost:58083 (size: 1209.0 B, free: 265.4 MB)
16/04/16 15:46:15 INFO SparkContext: Created broadcast 11 from broadcast at DAGScheduler.scala:874
16/04/16 15:46:15 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 10 (MapPartitionsRDD[15] at filter at <console>:19)
16/04/16 15:46:15 INFO TaskSchedulerImpl: Adding task set 10.0 with 2 tasks
16/04/16 15:46:15 INFO BlockManagerInfo: Removed broadcast_10_piece0 on localhost:58083 in memory (size: 1207.0 B, free: 265.4 MB)
16/04/16 15:46:15 INFO TaskSetManager: Starting task 0.0 in stage 10.0 (TID 29, localhost, PROCESS_LOCAL, 1369 bytes)
16/04/16 15:46:15 INFO TaskSetManager: Starting task 1.0 in stage 10.0 (TID 30, localhost, PROCESS_LOCAL, 1426 bytes)
16/04/16 15:46:15 INFO Executor: Running task 0.0 in stage 10.0 (TID 29)
16/04/16 15:46:15 INFO Executor: Running task 1.0 in stage 10.0 (TID 30)
16/04/16 15:46:15 INFO Executor: Finished task 1.0 in stage 10.0 (TID 30). 5626 bytes result sent to driver
16/04/16 15:46:15 INFO TaskSetManager: Finished task 1.0 in stage 10.0 (TID 30) in 17 ms on localhost (1/2)
16/04/16 15:46:15 INFO Executor: Finished task 0.0 in stage 10.0 (TID 29). 5626 bytes result sent to driver
16/04/16 15:46:15 INFO TaskSetManager: Finished task 0.0 in stage 10.0 (TID 29) in 45 ms on localhost (2/2)
16/04/16 15:46:15 INFO DAGScheduler: ResultStage 10 (collect at <console>:22) finished in 0.045 s
16/04/16 15:46:15 INFO TaskSchedulerImpl: Removed TaskSet 10.0, whose tasks have all completed, from pool
16/04/16 15:46:15 INFO DAGScheduler: Job 10 finished: collect at <console>:22, took 0.089623 s
res10: Array[Int] = Array(2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68,
70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128, 130, 132, 134, 136, 138
, 140, 142, 144, 146, 148, 150, 152, 154, 156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176, 178, 180, 182, 184, 186, 188, 190, 192, 194, 196, 198, 200, 2
02, 204, 206, 208, 210, 212, 214, 216, 218, 220, 222, 224, 226, 228, 230, 232, 234, 236, 238, 240, 242, 244, 246, 248, 250, 252, 254, 256, 258, 260, 262, 264,
 266, 268, 270, 272, 274, 276, 278, 280, 282, 284, 286, 288, 290, 292, 294, 296, 298, 300, 302, 304, 306, 308, 310, 312, 314, 316, 318, 320, 322, 324, 326, 32
8, 330, ...
```

Example: Scala example of map

```
val input = sc.parallelize(List(1, 2, 3, 4))

val result = input.map(x => x * x)

println(result.collect().mkString(","))
```

```
scala> val input = sc.parallelize(List(1, 2, 3, 4))
input: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[16] at parallelize at <console>:15

scala> val result = input.map(x => x * x)
result: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[17] at map at <console>:17
```

```
scala> println(result.collect().mkString(","))
16/04/16 15:47:45 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 15:47:45 INFO DAGScheduler: Got job 11 (collect at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 15:47:45 INFO DAGScheduler: Final stage: ResultStage 11(collect at <console>:20)
16/04/16 15:47:45 INFO DAGScheduler: Parents of final stage: List()
16/04/16 15:47:45 INFO DAGScheduler: Missing parents: List()
16/04/16 15:47:45 INFO DAGScheduler: Submitting ResultStage 11 (MapPartitionsRDD[17] at map at <console>:17), which has
16/04/16 15:47:45 INFO MemoryStore: ensureFreeSpace(1944) called with curMem=221565, maxMem=278302556
16/04/16 15:47:45 INFO MemoryStore: Block broadcast_12 stored as values in memory (estimated size 1944.0 B, free 265.2
16/04/16 15:47:45 INFO MemoryStore: ensureFreeSpace(1212) called with curMem=223509, maxMem=278302556
16/04/16 15:47:45 INFO MemoryStore: Block broadcast_12_piece0 stored as bytes in memory (estimated size 1212.0 B, free
16/04/16 15:47:45 INFO BlockManagerInfo: Added broadcast_12_piece0 in memory on localhost:58083 (size: 1212.0 B, free:
16/04/16 15:47:45 INFO SparkContext: Created broadcast 12 from broadcast at DAGScheduler.scala:874
16/04/16 15:47:45 INFO DAGScheduler: Submitting 4 missing tasks from ResultStage 11 (MapPartitionsRDD[17] at map at <co
16/04/16 15:47:45 INFO TaskSchedulerImpl: Adding task set 11.0 with 4 tasks
16/04/16 15:47:45 INFO TaskSetManager: Starting task 0.0 in stage 11.0 (TID 31, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:47:45 INFO TaskSetManager: Starting task 1.0 in stage 11.0 (TID 32, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:47:45 INFO TaskSetManager: Starting task 2.0 in stage 11.0 (TID 33, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:47:45 INFO TaskSetManager: Starting task 3.0 in stage 11.0 (TID 34, localhost, PROCESS_LOCAL, 1313 bytes)
16/04/16 15:47:45 INFO Executor: Running task 0.0 in stage 11.0 (TID 31)
16/04/16 15:47:45 INFO Executor: Running task 1.0 in stage 11.0 (TID 32)
16/04/16 15:47:45 INFO Executor: Running task 2.0 in stage 11.0 (TID 33)
16/04/16 15:47:45 INFO Executor: Running task 3.0 in stage 11.0 (TID 34)
16/04/16 15:47:45 INFO Executor: Finished task 0.0 in stage 11.0 (TID 31). 607 bytes result sent to driver
16/04/16 15:47:45 INFO Executor: Finished task 1.0 in stage 11.0 (TID 32). 607 bytes result sent to driver
16/04/16 15:47:45 INFO TaskSetManager: Finished task 1.0 in stage 11.0 (TID 32) in 9 ms on localhost (1/4)
16/04/16 15:47:45 INFO Executor: Finished task 3.0 in stage 11.0 (TID 34). 607 bytes result sent to driver
16/04/16 15:47:45 INFO TaskSetManager: Finished task 0.0 in stage 11.0 (TID 31) in 10 ms on localhost (2/4)
16/04/16 15:47:45 INFO TaskSetManager: Finished task 3.0 in stage 11.0 (TID 34) in 9 ms on localhost (3/4)
16/04/16 15:47:45 INFO Executor: Finished task 2.0 in stage 11.0 (TID 33). 607 bytes result sent to driver
16/04/16 15:47:45 INFO TaskSetManager: Finished task 2.0 in stage 11.0 (TID 33) in 12 ms on localhost (4/4)
16/04/16 15:47:45 INFO TaskSchedulerImpl: Removed TaskSet 11.0, whose tasks have all completed, from pool
16/04/16 15:47:45 INFO DAGScheduler: ResultStage 11 (collect at <console>:20) finished in 0.013 s
16/04/16 15:47:45 INFO DAGScheduler: Job 11 finished: collect at <console>:20, took 0.022441 s
1,4,9,16
```

Example 6: union

```
val seta = sc.parallelize(1 to 10)
val setb = sc.parallelize(5 to 15)
(seta union setb).collect
```

```
scala> val seta = sc.parallelize(1 to 10)
16/04/16 15:58:07 INFO BlockManagerInfo: Removed broadcast_12_piece0 on localhost:58883 in memory (size: 1212.0 B, free: 265.4 MB)
16/04/16 15:58:07 INFO BlockManagerInfo: Removed broadcast_11_piece0 on localhost:58883 in memory (size: 1209.0 B, free: 265.4 MB)
seta: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[18] at parallelize at <console>:15

scala> val setb = sc.parallelize(5 to 15)
setb: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[19] at parallelize at <console>:15
```

```
scala> (seta union setb).collect
16/04/16 15:50:40 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 15:50:40 INFO DAGScheduler: Got job 12 (collect at <console>:20) with 8 output partitions (allowLocal=false)

16/04/16 15:50:40 INFO TaskSetManager: Finished task 6.0 in stage 12.0 (TID 41) in 17 ms on localhost (6/8)
16/04/16 15:50:40 INFO TaskSetManager: Finished task 7.0 in stage 12.0 (TID 42) in 10 ms on localhost (7/8)
16/04/16 15:50:40 INFO TaskSetManager: Finished task 5.0 in stage 12.0 (TID 40) in 27 ms on localhost (8/8)
16/04/16 15:50:40 INFO TaskSchedulerImpl: Removed TaskSet 12.0, whose tasks have all completed, from pool
16/04/16 15:50:40 INFO DAGScheduler: ResultStage 12 (collect at <console>:20) finished in 0.047 s
16/04/16 15:50:40 INFO DAGScheduler: Job 12 finished: collect at <console>:20, took 0.071821 s
res12: Array[Int] = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)
```

Example 7: flatMap() using scala shell

Splitting lines into multiple words

val lines = sc.parallelize(List("hello world", "hi"))

val words = lines.flatMap(line => line.split(" "))

words.first() // returns "hello"

```
scala> val lines = sc.parallelize(List("hello world", "hi"))
lines: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[21] at parallelize at <console>:15

scala> val words = lines.flatMap(line => line.split(" "))
words: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[22] at flatMap at <console>:17
```

```
scala> words.first()
16/04/16 15:54:16 INFO SparkContext: Starting job: first at <console>:20
16/04/16 15:54:16 INFO DAGScheduler: Got job 13 (first at <console>:20) with 1 output partitions (allowLocal=true)
16/04/16 15:54:16 INFO DAGScheduler: Final stage: ResultStage 13(first at <console>:20)

16/04/16 15:54:16 INFO TaskSetManager: Finished task 2.0 in stage 14.0 (TID 46) in 22 ms on localhost (2/3)
16/04/16 15:54:16 INFO TaskSetManager: Finished task 1.0 in stage 14.0 (TID 45) in 25 ms on localhost (3/3)
16/04/16 15:54:16 INFO TaskSchedulerImpl: Removed TaskSet 14.0, whose tasks have all completed, from pool
16/04/16 15:54:16 INFO DAGScheduler: ResultStage 14 (first at <console>:20) finished in 0.021 s
16/04/16 15:54:16 INFO DAGScheduler: Job 14 finished: first at <console>:20, took 0.063394 s
res13: String = hello
```

3. Paired RDD

- Spark provides special operations on RDDs containing key/value pairs. These RDDs are called pair RDD.
- Pair RDD allows you to act on each key in parallel or regroup data across the network.

Example 8: foldByKey

```
val a = sc.parallelize(List("kim","kumar","muthu","tim","lak","vamsi"),2)
val b = a.map(x => (x.length,x))

b.collect
b.foldByKey("")(_+_).collect
```

```
scala> val a = sc.parallelize(List("kim","kumar","muthu","tim","lak","vamsi"),2)
a: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[24] at parallelize at <console>:15

scala> val b = a.map(x => (x.length,x))
b: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[25] at map at <console>:17
```

```
scala> b.collect
16/04/16 16:04:40 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:04:40 INFO DAGScheduler: Got job 15 (collect at <console>:20) with 2 output partitions (allowLocal=false)
16/04/16 16:04:40 INFO DAGScheduler: Final stage: ResultStage 15(collect at <console>:20)
16/04/16 16:04:40 INFO DAGScheduler: Parents of final stage: List()
16/04/16 16:04:40 INFO DAGScheduler: Missing parents: List()
16/04/16 16:04:40 INFO DAGScheduler: Submitting ResultStage 15 (MapPartitionsRDD[25] at map at <console>:17), which has no missing
16/04/16 16:04:40 INFO MemoryStore: ensureFreeSpace(1912) called with curMem=218412, maxMem=278302556
16/04/16 16:04:40 INFO MemoryStore: Block broadcast_16 stored as values in memory (estimated size 1912.0 B, free 265.2 MB)
16/04/16 16:04:40 INFO MemoryStore: ensureFreeSpace(1185) called with curMem=220324, maxMem=278302556
16/04/16 16:04:40 INFO MemoryStore: Block broadcast_16_piece0 stored as bytes in memory (estimated size 1185.0 B, free 265.2 MB)
16/04/16 16:04:40 INFO BlockManagerInfo: Added broadcast_16_piece0 in memory on localhost:50083 (size: 1185.0 B, free: 265.4 MB)
16/04/16 16:04:40 INFO SparkContext: Created broadcast 16 from broadcast at DAGScheduler.scala:874
16/04/16 16:04:40 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 15 (MapPartitionsRDD[25] at map at <console>:17)
16/04/16 16:04:40 INFO TaskSchedulerImpl: Adding task set 15.0 with 2 tasks
16/04/16 16:04:40 INFO TaskSetManager: Starting task 0.0 in stage 15.0 (TID 47, localhost, PROCESS_LOCAL, 1393 bytes)
16/04/16 16:04:40 INFO TaskSetManager: Starting task 1.0 in stage 15.0 (TID 48, localhost, PROCESS_LOCAL, 1391 bytes)
16/04/16 16:04:40 INFO Executor: Running task 0.0 in stage 15.0 (TID 47)
16/04/16 16:04:40 INFO Executor: Running task 1.0 in stage 15.0 (TID 48)
16/04/16 16:04:40 INFO Executor: Finished task 1.0 in stage 15.0 (TID 48). 885 bytes result sent to driver
16/04/16 16:04:40 INFO Executor: Finished task 0.0 in stage 15.0 (TID 47). 887 bytes result sent to driver
16/04/16 16:04:40 INFO TaskSetManager: Finished task 1.0 in stage 15.0 (TID 48) in 6 ms on localhost (1/2)
16/04/16 16:04:40 INFO TaskSetManager: Finished task 0.0 in stage 15.0 (TID 47) in 8 ms on localhost (2/2)
16/04/16 16:04:40 INFO DAGScheduler: ResultStage 15 (collect at <console>:20) finished in 0.009 s
16/04/16 16:04:40 INFO TaskSchedulerImpl: Removed TaskSet 15.0, whose tasks have all completed, from pool
16/04/16 16:04:40 INFO DAGScheduler: Job 15 finished: collect at <console>:20, took 0.030737 s
res14: Array[(Int, String)] = Array((3,kim), (5,kumar), (5,muthu), (3,tim), (3,lak), (5,vamsi))
```

```
scala> b.foldByKey("")(_+_).collect
16/04/16 16:05:45 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:05:45 INFO DAGScheduler: Registering RDD 25 (map at <console>:17)
16/04/16 16:05:45 INFO DAGScheduler: Got job 16 (collect at <console>:20) with 2 output partitions (allowLocal=false)
16/04/16 16:05:45 INFO DAGScheduler: Final stage: ResultStage 17(collect at <console>:20)

16/04/16 16:05:45 INFO Executor: Finished task 0.0 in stage 17.0 (TID 51). 882 bytes result sent to driver
16/04/16 16:05:45 INFO Executor: Finished task 1.0 in stage 17.0 (TID 52). 1070 bytes result sent to driver
16/04/16 16:05:45 INFO TaskSetManager: Finished task 0.0 in stage 17.0 (TID 51) in 60 ms on localhost (1/2)
16/04/16 16:05:45 INFO TaskSetManager: Finished task 1.0 in stage 17.0 (TID 52) in 59 ms on localhost (2/2)
16/04/16 16:05:45 INFO TaskSchedulerImpl: Removed TaskSet 17.0, whose tasks have all completed, from pool
16/04/16 16:05:45 INFO DAGScheduler: ResultStage 17 (collect at <console>:20) finished in 0.061 s
16/04/16 16:05:45 INFO DAGScheduler: Job 16 finished: collect at <console>:20, took 0.167241 s
res15: Array[(Int, String)] = Array((3,kimtimlak), (5,kumarmuthuvamsi))
```

Example 9: foldByKey

```
val deptEmployees =
  List
  (
   ("dept1",("kumar1",1000.0)),
   ("dept1",("kumar2",1200.0)),
   ("dept2",("kumar3",2200.0)),
   ("dept2",("kumar4",1400.0)),
   ("dept2",("kumar5",1000.0)),
   ("dept2",("kumar6",800.0)),
   ("dept1",("kumar7",2000.0)),
   ("dept1",("kumar8",1000.0)),
   ("dept1",("kumar9",500.0))
   )


val employeeRDD = sc.makeRDD(deptEmployees)
val maxByDept = employeeRDD.foldByKey(("dummy",Double.MinValue))((acc,element)
=> if(acc._2 > element._2)acc else element)

println("Maximum salaries in each dept" + maxByDept.collect().toList)
```

```
scala> val deptEmployees = List (("dept1",("kumar1",1000.0)),
     | ("dept1",("kumar2",1200.0)),
     | ("dept2",("kumar3",2200.0)),
     | ("dept2",("kumar4",1400.0)),
     | ("dept2",("kumar5",1000.0)),
     | ("dept2",("kumar6",800.0)),
     | ("dept1",("kumar7",2000.0)),
     | ("dept1",("kumar8",1000.0)),
     | ("dept1",("kumar9",500.0))
     | )
deptEmployees: List[(String, (String, Double))] = List((dept1,(kumar1,1000.0)), (dept1,(kumar2,1200.0)), (dept2,(kumar3,2200.0)), (dept2,(kumar4,1400.0)), (de
pt2,(kumar5,1000.0)), (dept2,(kumar6,800.0)), (dept1,(kumar7,2000.0)), (dept1,(kumar8,1000.0)), (dept1,(kumar9,500.0)))
```

```
scala> val employeeRDD = sc.makeRDD(deptEmployees)
employeeRDD: org.apache.spark.rdd.RDD[(String, (String, Double))] = ParallelCollectionRDD[27] at makeRDD at <console>:17
```

```
scala> val maxByDept = employeeRDD.foldByKey(("dummy",Double.MinValue))((acc,element)=>if(acc._2 > element._2)acc else element)
maxByDept: org.apache.spark.rdd.RDD[(String, (String, Double))] = ShuffledRDD[29] at foldByKey at <console>:19
```

```
scala> println("Maximum salaries in eachdept" + maxByDept.collect().toList)
16/04/16 16:13:51 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 16:13:51 INFO DAGScheduler: Registering RDD 27 (makeRDD at <console>:17)
16/04/16 16:13:51 INFO DAGScheduler: Got job 17 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 16:13:51 INFO DAGScheduler: Final stage: ResultStage 19(collect at <console>:22)
16/04/16 16:13:51 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 18)
16/04/16 16:13:51 INFO DAGScheduler: Missing parents: List(ShuffleMapStage 18)
```

```
16/04/16 16:13:51 INFO Executor: Finished task 3.0 in stage 19.0 (TID 60). 882 bytes result sent to driver
16/04/16 16:13:51 INFO TaskSetManager: Finished task 3.0 in stage 19.0 (TID 60) in 12 ms on localhost (3/4)
16/04/16 16:13:51 INFO ShuffleBlockFetcherIterator: Getting 2 non-empty blocks out of 4 blocks
16/04/16 16:13:51 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
16/04/16 16:13:51 INFO Executor: Finished task 0.0 in stage 19.0 (TID 57). 1050 bytes result sent to driver
16/04/16 16:13:51 INFO TaskSetManager: Finished task 0.0 in stage 19.0 (TID 57) in 35 ms on localhost (4/4)
16/04/16 16:13:51 INFO TaskSchedulerImpl: Removed TaskSet 19.0, whose tasks have all completed, from pool
16/04/16 16:13:51 INFO DAGScheduler: ResultStage 19 (collect at <console>:22) finished in 0.035 s
16/04/16 16:13:51 INFO DAGScheduler: Job 17 finished: collect at <console>:22, took 0.086547 s
Maximum salaries in eachdeptList((dept1,(kumar7,2000.0)), (dept2,(kumar3,2200.0)))
```

Example 10: reduceByKey

```
val textFile = sc.textFile("file:/home/technocrafty/Datasets/sample.txt ")

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

```
scala> val counts = textFile.flatMap(line => line.split(" "))
counts: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[30] at flatMap at <console>:17
```

```
scala> counts.collect
16/04/16 16:17:16 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:17:16 INFO DAGScheduler: Got job 18 (collect at <console>:20) with 2 output partitions (allowLocal=false)
16/04/16 16:17:16 INFO DAGScheduler: Final stage: ResultStage 20(collect at <console>:20)
16/04/16 16:17:16 INFO DAGScheduler: Parents of final stage: List()
16/04/16 16:17:16 INFO DAGScheduler: Missing parents: List()
16/04/16 16:17:16 INFO DAGScheduler: Submitting ResultStage 20 (MapPartitionsRDD[30] at flatMap at <console>:17), which has no missing parents
16/04/16 16:17:16 INFO MemoryStore: ensureFreeSpace(3368) called with curMem=229229, maxMem=278302556
16/04/16 16:17:16 INFO MemoryStore: Block broadcast_21 stored as values in memory (estimated size 3.3 KB, free 265.2 MB)
16/04/16 16:17:16 INFO MemoryStore: ensureFreeSpace(1927) called with curMem=232597, maxMem=278302556
16/04/16 16:17:16 INFO MemoryStore: Block broadcast_21_piece0 stored as bytes in memory (estimated size 1927.0 B, free 265.2 MB)
16/04/16 16:17:16 INFO BlockManagerInfo: Added broadcast_21_piece0 in memory on localhost:50083 (size: 1927.0 B, free: 265.4 MB)
16/04/16 16:17:16 INFO SparkContext: Created broadcast 21 from broadcast at DAGScheduler.scala:874
16/04/16 16:17:16 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 20 (MapPartitionsRDD[30] at flatMap at <console>:17)
16/04/16 16:17:16 INFO TaskSchedulerImpl: Adding task set 20.0 with 2 tasks
16/04/16 16:17:16 INFO TaskSetManager: Starting task 0.0 in stage 20.0 (TID 61, localhost, PROCESS_LOCAL, 1416 bytes)
16/04/16 16:17:16 INFO TaskSetManager: Starting task 1.0 in stage 20.0 (TID 62, localhost, PROCESS_LOCAL, 1416 bytes)
16/04/16 16:17:16 INFO Executor: Running task 0.0 in stage 20.0 (TID 61)
16/04/16 16:17:16 INFO Executor: Running task 1.0 in stage 20.0 (TID 62)
16/04/16 16:17:16 INFO HadoopRDD: Input split: file:/home/technocrafty/Datasets/sample.txt:0+255
16/04/16 16:17:16 INFO HadoopRDD: Input split: file:/home/technocrafty/Datasets/sample.txt:255+256
16/04/16 16:17:16 INFO Executor: Finished task 1.0 in stage 20.0 (TID 62). 1792 bytes result sent to driver
16/04/16 16:17:16 INFO Executor: Finished task 0.0 in stage 20.0 (TID 61). 2464 bytes result sent to driver
16/04/16 16:17:16 INFO TaskSetManager: Finished task 1.0 in stage 20.0 (TID 62) in 13 ms on localhost (1/2)
16/04/16 16:17:16 INFO TaskSetManager: Finished task 0.0 in stage 20.0 (TID 61) in 20 ms on localhost (2/2)
16/04/16 16:17:16 INFO DAGScheduler: ResultStage 20 (collect at <console>:20) finished in 0.020 s
16/04/16 16:17:16 INFO DAGScheduler: Job 18 finished: collect at <console>:20, took 0.039580 s
16/04/16 16:17:16 INFO TaskSchedulerImpl: Removed TaskSet 20.0, whose tasks have all completed, from pool
res17: Array[String] = Array(Apache, Hadoop, is, an, open, source, framework, for, distributed, storage, and, processing, of, large, sets, of, data, on, commo
dity, hardware., Hadoop, enables, businesses, to, quickly, gain, insight, from, massive, amounts, of, structured, and, unstructured, data., "", Numerous, Apac
he, Software, Foundation, projects, make, up, the, services, required, by, an, enterprise, to, deploy,, integrate, and, work, with, Hadoop., "", Each, project
, has, been, developed, to, deliver, an, explicit, function, and, each, has, its, own, community, of, developers, and, individual, release, cycles.)
```

```
scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word,1))
counts: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[32] at map at <console>:17
```

```
scala> counts.collect
16/04/16 16:19:27 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:19:27 INFO DAGScheduler: Got job 19 (collect at <console>:20) with 2 output partitions (allowLocal=false)
16/04/16 16:19:27 INFO DAGScheduler: Final stage: ResultStage 21(collect at <console>:20)
16/04/16 16:19:27 INFO DAGScheduler: Parents of final stage: List()
16/04/16 16:19:27 INFO DAGScheduler: Missing parents: List()
16/04/16 16:19:27 INFO DAGScheduler: Submitting ResultStage 21 (MapPartitionsRDD[32] at map at <console>:17), which has no missing parents
16/04/16 16:19:27 INFO MemoryStore: ensureFreeSpace(3504) called with curMem=234524, maxMem=278302556
16/04/16 16:19:27 INFO MemoryStore: Block broadcast_22 stored as values in memory (estimated size 3.4 KB, free 265.2 MB)
16/04/16 16:19:27 INFO MemoryStore: ensureFreeSpace(1962) called with curMem=238028, maxMem=278302556
16/04/16 16:19:27 INFO MemoryStore: Block broadcast_22_piece0 stored as bytes in memory (estimated size 1962.0 B, free 265.2 MB)
16/04/16 16:19:27 INFO BlockManagerInfo: Added broadcast_22_piece0 in memory on localhost:58083 (size: 1962.0 B, free: 265.4 MB)
16/04/16 16:19:27 INFO SparkContext: Created broadcast 22 from broadcast at DAGScheduler.scala:874
16/04/16 16:19:27 INFO DAGScheduler: Submitting 2 missing tasks from ResultStage 21 (MapPartitionsRDD[32] at map at <console>:17)
16/04/16 16:19:27 INFO TaskSchedulerImpl: Adding task set 21.0 with 2 tasks
16/04/16 16:19:27 INFO TaskSetManager: Starting task 0.0 in stage 21.0 (TID 63, localhost, PROCESS_LOCAL, 1416 bytes)
16/04/16 16:19:27 INFO TaskSetManager: Starting task 1.0 in stage 21.0 (TID 64, localhost, PROCESS_LOCAL, 1416 bytes)
16/04/16 16:19:27 INFO Executor: Running task 0.0 in stage 21.0 (TID 63)
16/04/16 16:19:27 INFO Executor: Running task 1.0 in stage 21.0 (TID 64)
16/04/16 16:19:27 INFO HadoopRDD: Input split: file:/home/technocrafty/Datasets/sample.txt:0+255
16/04/16 16:19:27 INFO Executor: Finished task 0.0 in stage 21.0 (TID 63). 3465 bytes result sent to driver
16/04/16 16:19:27 INFO TaskSetManager: Finished task 0.0 in stage 21.0 (TID 63) in 12 ms on localhost (1/2)
16/04/16 16:19:27 INFO HadoopRDD: Input split: file:/home/technocrafty/Datasets/sample.txt:255+256
16/04/16 16:19:27 INFO Executor: Finished task 1.0 in stage 21.0 (TID 64). 1788 bytes result sent to driver
16/04/16 16:19:27 INFO TaskSetManager: Finished task 1.0 in stage 21.0 (TID 64) in 28 ms on localhost (2/2)
16/04/16 16:19:27 INFO TaskSchedulerImpl: Removed TaskSet 21.0, whose tasks have all completed, from pool
16/04/16 16:19:27 INFO DAGScheduler: ResultStage 21 (collect at <console>:20) finished in 0.029 s
16/04/16 16:19:27 INFO DAGScheduler: Job 19 finished: collect at <console>:20, took 0.040238 s
res18: Array[(String, Int)] = Array((Apache,1), (Hadoop,1), (is,1), (an,1), (open,1), (source,1), (framework,1), (for,1), (distributed,1), (storage,1), (and,1
), (processing,1), (of,1), (large,1), (sets,1), (of,1), (data,1), (on,1), (commodity,1), (hardware.,1), (Hadoop,1), (enables,1), (businesses,1), (to,1), (quic
kly,1), (gain,1), (insight,1), (from,1), (massive,1), (amounts,1), (of,1), (structured,1), (and,1), (unstructured,1), (data.,1), ("",1), (Numerous,1), (Apache
,1), (Software,1), (Foundation,1), (projects,1), (make,1), (up,1), (the,1), (services,1), (required,1), (by,1), (an,1), (enterprise,1), (to,1), (deploy,,1), (
integrate,1), (and,1), (work,1), (with,1), (Hadoop.,1), ("",1), (Each,1), (project,1), (has,1), (been,1), (developed,1), (to,1), (deliver,1), (an,1), (explici
t,1), (_,...
```

```
scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word,1)).reduceByKey(_+_)
16/04/16 16:20:40 INFO BlockManagerInfo: Removed broadcast_22_piece0 on localhost:58083 in memory (size: 1962.0 B, free: 265.4 MB)
16/04/16 16:20:40 INFO BlockManagerInfo: Removed broadcast_21_piece0 on localhost:58083 in memory (size: 1927.0 B, free: 265.4 MB)
16/04/16 16:20:40 INFO BlockManagerInfo: Removed broadcast_20_piece0 on localhost:58083 in memory (size: 2.0 KB, free: 265.4 MB)
16/04/16 16:20:40 INFO BlockManagerInfo: Removed broadcast_19_piece0 on localhost:58083 in memory (size: 1737.0 B, free: 265.4 MB)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[35] at reduceByKey at <console>:17
```

```
scala> counts.collect.foreach(println)
16/04/16 16:21:24 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:21:24 INFO DAGScheduler: Registering RDD 34 (map at <console>:17)
16/04/16 16:21:24 INFO DAGScheduler: Got job 20 (collect at <console>:20) with 2 output partitions (allowLocal=false)
16/04/16 16:21:24 INFO DAGScheduler: Final stage: ResultStage 23(collect at <console>:20)
16/04/16 16:21:24 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 22)
```

Final print output logs are too big to display here, but the content will look as below

```
(integrate,1)
(by,1)
(individual,1)
(storage,1)
(structured,1)
(for,1)
(Each,1)
(amounts,1)
(cycles.,1)
(an,3)
(Foundation,1)
(distributed,1)
(and,5)
(required,1)
(deploy,,1)
(the,1)
(Hadoop,2)

scala>
```

Example 11: reduceByKey

val myRDD = sc.parallelize(Seq((1,"A"),(2,"B"),(2,"D"),(3,"C"),(3,"A"),(3,"B"),(3,"A")),1)
val resultRDD = myRDD.reduceByKey((x, y) => {println("x"+x+"::"+"y"+y);x+y})
resultRDD.foreach(println)

```
scala> val myRDD =sc.parallelize(Seq((1,"A"),(2,"B"),(2,"D"),(3,"C"),(3,"A"),(3,"B"),(3,"A")),1)
myRDD: org.apache.spark.rdd.RDD[(Int, String)] = ParallelCollectionRDD[38] at parallelize at <console>:15

scala> val resultRDD = myRDD.reduceByKey((x,y) => {println("x"+x+"::"+"y"+y);x+y})
resultRDD: org.apache.spark.rdd.RDD[(Int, String)] = ShuffledRDD[39] at reduceByKey at <console>:17
```

```
scala> resultRDD.foreach(println)
16/04/16 16:25:03 INFO SparkContext: Starting job: foreach at <console>:20
16/04/16 16:25:03 INFO DAGScheduler: Registering RDD 38 (parallelize at <console>:15)
16/04/16 16:25:03 INFO DAGScheduler: Got job 21 (foreach at <console>:20) with 1 output partitions (allowLocal=false)
```

```
16/04/16 16:25:04 INFO ShuffleBlockFetcherIterator: Getting 1 non-empty blocks out of 1 blocks
16/04/16 16:25:04 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 0 ms
(1,A)
(3,CABA)
(2,BD)
16/04/16 16:25:04 INFO Executor: Finished task 0.0 in stage 25.0 (TID 70). 886 bytes result sent to driver
16/04/16 16:25:04 INFO TaskSetManager: Finished task 0.0 in stage 25.0 (TID 70) in 6 ms on localhost (1/1)
16/04/16 16:25:04 INFO TaskSchedulerImpl: Removed TaskSet 25.0, whose tasks have all completed, from pool
16/04/16 16:25:04 INFO DAGScheduler: ResultStage 25 (foreach at <console>:20) finished in 0.007 s
16/04/16 16:25:04 INFO DAGScheduler: Job 21 finished: foreach at <console>:20, took 0.027969 s
```

Example 12: collectAsMap

val myRDD = sc.parallelize(Seq((1,"A"),(2,"B"),(2,"D"),(3,"C"),(3,"A"),(3,"B"),

(3,"A")))
myRDD.collectAsMap()

```
scala> val myRDD = sc.parallelize(Seq((1,"A"),(2,"B"),(2,"D"),(3,"C"),(3,"A"),(3,"B"),
     | (3,"A")))
myRDD: org.apache.spark.rdd.RDD[(Int, String)] = ParallelCollectionRDD[40] at parallelize at <console>:15

scala> myRDD.collectAsMap()
16/04/16 16:27:37 INFO SparkContext: Starting job: collectAsMap at <console>:18
16/04/16 16:27:37 INFO DAGScheduler: Got job 22 (collectAsMap at <console>:18) with 4 output partitions (allowLocal=false)
16/04/16 16:27:37 INFO DAGScheduler: Final stage: ResultStage 26(collectAsMap at <console>:18)
16/04/16 16:27:37 INFO DAGScheduler: Parents of final stage: List()
```

```
16/04/16 16:27:37 INFO Executor: Running task 3.0 in stage 26.0 (TID 74)
16/04/16 16:27:37 INFO TaskSetManager: Finished task 0.0 in stage 26.0 (TID 71) in 18 ms on localhost (1/4)
16/04/16 16:27:37 INFO TaskSetManager: Finished task 2.0 in stage 26.0 (TID 73) in 17 ms on localhost (2/4)
16/04/16 16:27:37 INFO TaskSetManager: Finished task 1.0 in stage 26.0 (TID 72) in 18 ms on localhost (3/4)
16/04/16 16:27:37 INFO Executor: Finished task 3.0 in stage 26.0 (TID 74). 777 bytes result sent to driver
16/04/16 16:27:37 INFO BlockManagerInfo: Removed broadcast_26_piece0 on localhost:58883 in memory (size: 1346.0 B, free: 265.4 MB)
16/04/16 16:27:37 INFO TaskSetManager: Finished task 3.0 in stage 26.0 (TID 74) in 20 ms on localhost (4/4)
16/04/16 16:27:37 INFO DAGScheduler: ResultStage 26 (collectAsMap at <console>:18) finished in 0.023 s
16/04/16 16:27:37 INFO TaskSchedulerImpl: Removed TaskSet 26.0, whose tasks have all completed, from pool
16/04/16 16:27:37 INFO DAGScheduler: Job 22 finished: collectAsMap at <console>:18, took 0.049899 s
res21: scala.collection.Map[Int,String] = Map(2 -> D, 1 -> A, 3 -> A)
```

Example 13: countByKey

myRDD.countByKey()

```
scala> myRDD.countByKey()
16/04/16 16:28:48 INFO SparkContext: Starting job: countByKey at <console>:18
16/04/16 16:28:48 INFO DAGScheduler: Registering RDD 41 (countByKey at <console>:18)
16/04/16 16:28:48 INFO DAGScheduler: Got job 23 (countByKey at <console>:18) with 4 output partitions (allowLocal=false)
16/04/16 16:28:48 INFO DAGScheduler: Final stage: ResultStage 28(countByKey at <console>:18)
```

```
16/04/16 16:28:40 INFO TaskSetManager: Finished task 0.0 in stage 28.0 (TID 79) in 13 ms on localhost (2/4)
16/04/16 16:28:40 INFO TaskSetManager: Finished task 3.0 in stage 28.0 (TID 82) in 12 ms on localhost (3/4)
16/04/16 16:28:40 INFO Executor: Finished task 1.0 in stage 28.0 (TID 80). 1075 bytes result sent to driver
16/04/16 16:28:40 INFO TaskSetManager: Finished task 1.0 in stage 28.0 (TID 80) in 15 ms on localhost (4/4)
16/04/16 16:28:40 INFO TaskSchedulerImpl: Removed TaskSet 28.0, whose tasks have all completed, from pool
16/04/16 16:28:40 INFO DAGScheduler: ResultStage 28 (countByKey at <console>:18) finished in 0.015 s
16/04/16 16:28:40 INFO DAGScheduler: Job 23 finished: countByKey at <console>:18, took 0.057667 s
res22: scala.collection.Map[Int,Long] = Map(1 -> 1, 2 -> 2, 3 -> 4)
```

Example 14: groupBy

```
val a = sc.parallelize(1 to 15)
a.groupBy(x => {if (x % 2 == 0) "even" else"odd"}).collect
```

```
scala> val a = sc.parallelize(1 to 15)
a: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[0] at parallelize at <console>:15

scala> a.groupBy(x => {if (x % 2 == 0) "even" else"odd"}).collect
16/04/16 16:30:52 INFO SparkContext: Starting job: collect at <console>:18
16/04/16 16:30:52 INFO DAGScheduler: Registering RDD 1 (groupBy at <console>:18)
16/04/16 16:30:52 INFO DAGScheduler: Got job 0 (collect at <console>:18) with 4 output partitions (allowLocal=false)
```

```
16/04/16 16:30:53 INFO TaskSetManager: Finished task 0.0 in stage 1.0 (TID 4) in 98 ms on localhost (1/4)
16/04/16 16:30:53 INFO TaskSetManager: Finished task 1.0 in stage 1.0 (TID 5) in 98 ms on localhost (2/4)
16/04/16 16:30:53 INFO Executor: Finished task 2.0 in stage 1.0 (TID 6). 1467 bytes result sent to driver
16/04/16 16:30:53 INFO Executor: Finished task 3.0 in stage 1.0 (TID 7). 1466 bytes result sent to driver
16/04/16 16:30:53 INFO TaskSetManager: Finished task 2.0 in stage 1.0 (TID 6) in 189 ms on localhost (3/4)
16/04/16 16:30:53 INFO TaskSetManager: Finished task 3.0 in stage 1.0 (TID 7) in 111 ms on localhost (4/4)
16/04/16 16:30:53 INFO TaskSchedulerImpl: Removed TaskSet 1.0, whose tasks have all completed, from pool
16/04/16 16:30:53 INFO DAGScheduler: ResultStage 1 (collect at <console>:18) finished in 0.100 s
16/04/16 16:30:53 INFO DAGScheduler: Job 0 finished: collect at <console>:18, took 0.520366 s
res0: Array[(String, Iterable[Int])] = Array((even,CompactBuffer(2, 4, 6, 8, 10, 12, 14)), (odd,CompactBuffer(1, 3, 5, 7, 9, 11, 13, 15)))
```

Example 15: groupByKey

```
val name = sc.parallelize(List("kim","kumar","muthu","tim","lak","vams"))
val namekey = name.keyBy(_.length)
val mycounter = namekey.map(x => (x._1,1))
mycounter.collect
```

```
scala> val name = sc.parallelize(List("kim","kumar","muthu","tim","lak","vams"))
name: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[3] at parallelize at <console>:15

scala> val namekey = name.keyBy(_.length)
namekey: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[4] at keyBy at <console>:17

scala> val mycounter = namekey.map(x => (x._1,1))
mycounter: org.apache.spark.rdd.RDD[(Int, Int)] = MapPartitionsRDD[5] at map at <console>:19

scala> mycounter.collect
16/04/16 16:32:30 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 16:32:30 INFO DAGScheduler: Got job 1 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 16:32:30 INFO DAGScheduler: Final stage: ResultStage 2(collect at <console>:22)
16/04/16 16:32:30 INFO DAGScheduler: Parents of final stage: List()
```

```
16/04/16 16:32:30 INFO Executor: Finished task 0.0 in stage 2.0 (TID 8). 750 bytes result sent to driver
16/04/16 16:32:30 INFO TaskSetManager: Finished task 0.0 in stage 2.0 (TID 8) in 24 ms on localhost (2/4)
16/04/16 16:32:30 INFO Executor: Finished task 3.0 in stage 2.0 (TID 11). 766 bytes result sent to driver
16/04/16 16:32:30 INFO TaskSetManager: Finished task 3.0 in stage 2.0 (TID 11) in 18 ms on localhost (3/4)
16/04/16 16:32:30 INFO Executor: Finished task 1.0 in stage 2.0 (TID 9). 766 bytes result sent to driver
16/04/16 16:32:30 INFO TaskSetManager: Finished task 1.0 in stage 2.0 (TID 9) in 22 ms on localhost (4/4)
16/04/16 16:32:30 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
16/04/16 16:32:30 INFO DAGScheduler: ResultStage 2 (collect at <console>:22) finished in 0.022 s
16/04/16 16:32:30 INFO DAGScheduler: Job 1 finished: collect at <console>:22, took 0.040170 s
res1: Array[(Int, Int)] = Array((3,1), (5,1), (5,1), (3,1), (3,1), (4,1))
```

Example 16: Join

val name = sc.parallelize(List("kim","kumar","muthu","tim","lak","vams"))
val namekey = name.keyBy(_.length)

namekey.collect

val sub = sc.parallelize(List("English","Maths","Tamil","Science"))

val subkey = sub.keyBy(_.length)

subkey.collect

namekey.join(subkey).collect

```
scala> val name = sc.parallelize(List("kim","kumar","muthu","tim","lak","vams"))
name: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[6] at parallelize at <console>:15

scala> val namekey = name.keyBy(_.length)
namekey: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[7] at keyBy at <console>:17

scala> namekey.collect
16/04/16 16:33:52 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:33:52 INFO DAGScheduler: Got job 2 (collect at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 16:33:52 INFO DAGScheduler: Final stage: ResultStage 3(collect at <console>:20)
```

```
16/04/16 16:33:52 INFO Executor: Finished task 2.0 in stage 3.0 (TID 14). 764 bytes result sent to driver
16/04/16 16:33:52 INFO Executor: Finished task 3.0 in stage 3.0 (TID 15). 787 bytes result sent to driver
16/04/16 16:33:52 INFO TaskSetManager: Finished task 2.0 in stage 3.0 (TID 14) in 21 ms on localhost (3/4)
16/04/16 16:33:52 INFO TaskSetManager: Finished task 3.0 in stage 3.0 (TID 15) in 22 ms on localhost (4/4)
16/04/16 16:33:52 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all completed, from pool
16/04/16 16:33:52 INFO DAGScheduler: ResultStage 3 (collect at <console>:20) finished in 0.032 s
16/04/16 16:33:52 INFO DAGScheduler: Job 2 finished: collect at <console>:20, took 0.104835 s
res2: Array[(Int, String)] = Array((3,kim), (5,kumar), (5,muthu), (3,tim), (3,lak), (4,vams))
```

```
scala> val sub = sc.parallelize(List ("English","Maths","Tamil","Science"))
sub: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[8] at parallelize at <console>:15

scala> val subkey = sub.keyBy(_.length)
subkey: org.apache.spark.rdd.RDD[(Int, String)] = MapPartitionsRDD[9] at keyBy at <console>:17

scala> subkey.collect
16/04/16 16:46:20 INFO SparkContext: Starting job: collect at <console>:20
16/04/16 16:46:20 INFO DAGScheduler: Got job 3 (collect at <console>:20) with 4 output partitions (allowLocal=false)
16/04/16 16:46:20 INFO DAGScheduler: Final stage: ResultStage 4(collect at <console>:20)
16/04/16 16:46:20 INFO DAGScheduler: Parents of final stage: List()
```

```
16/04/16 16:46:20 INFO TaskSetManager: Finished task 2.0 in stage 4.0 (TID 18) in 17 ms on localhost (2/4)
16/04/16 16:46:20 INFO TaskSetManager: Finished task 3.0 in stage 4.0 (TID 19) in 17 ms on localhost (3/4)
16/04/16 16:46:20 INFO Executor: Finished task 1.0 in stage 4.0 (TID 17). 766 bytes result sent to driver
16/04/16 16:46:20 INFO TaskSetManager: Finished task 1.0 in stage 4.0 (TID 17) in 22 ms on localhost (4/4)
16/04/16 16:46:20 INFO DAGScheduler: ResultStage 4 (collect at <console>:20) finished in 0.019 s
16/04/16 16:46:20 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks have all completed, from pool
16/04/16 16:46:20 INFO DAGScheduler: Job 3 finished: collect at <console>:20, took 0.035289 s
res3: Array[(Int, String)] = Array((7,English), (5,Maths), (5,Tamil), (7,Science))
```

```
scala> namekey.join(subkey).collect
16/04/16 16:47:29 INFO SparkContext: Starting job: collect at <console>:24
16/04/16 16:47:29 INFO DAGScheduler: Registering RDD 7 (keyBy at <console>:17)
16/04/16 16:47:29 INFO DAGScheduler: Registering RDD 9 (keyBy at <console>:17)
16/04/16 16:47:29 INFO DAGScheduler: Got job 4 (collect at <console>:24) with 4 output partitions (allowLocal=false)
16/04/16 16:47:29 INFO DAGScheduler: Final stage: ResultStage 7(collect at <console>:24)
16/04/16 16:47:29 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 5, ShuffleMapStage 6)
```

```
16/04/16 16:47:29 INFO TaskSetManager: Finished task 0.0 in stage 7.0 (TID 28) in 37 ms on localhost (2/4)
16/04/16 16:47:29 INFO Executor: Finished task 3.0 in stage 7.0 (TID 31). 882 bytes result sent to driver
16/04/16 16:47:29 INFO Executor: Finished task 1.0 in stage 7.0 (TID 29). 1136 bytes result sent to driver
16/04/16 16:47:29 INFO TaskSetManager: Finished task 3.0 in stage 7.0 (TID 31) in 57 ms on localhost (3/4)
16/04/16 16:47:29 INFO TaskSetManager: Finished task 1.0 in stage 7.0 (TID 29) in 60 ms on localhost (4/4)
16/04/16 16:47:29 INFO TaskSchedulerImpl: Removed TaskSet 7.0, whose tasks have all completed, from pool
16/04/16 16:47:29 INFO DAGScheduler: ResultStage 7 (collect at <console>:24) finished in 0.062 s
16/04/16 16:47:29 INFO DAGScheduler: Job 4 finished: collect at <console>:24, took 0.175332 s
res4: Array[(Int, (String, String))] = Array((5,(kumar,Maths)), (5,(kumar,Tamil)), (5,(muthu,Maths)), (5,(muthu,Tamil)))
```

Example 17: Join

Inner Join

val names1 = sc.parallelize(List("apple", "mango", "grapes")).map(a => (a,1))

val names2 = sc.parallelize(List("grapes", "litchi", "pears")).map(a => (a,1))

names1.join(names2).collect

leftOuterJoin

names1.leftOuterJoin(names2).collect

rightOuterJoin

names1.rightOuterJoin(names2).collect

```
scala> val names1 = sc.parallelize(List("apple", "mango", "grapes")).map(a => (a,1))
names1: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[14] at map at <console>:17

scala> val names2 = sc.parallelize(List("grapes", "litchi", "pears")).map(a => (a,1))
names2: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[16] at map at <console>:17

scala> names1.join(names2).collect
16/04/16 16:51:11 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 16:51:11 INFO DAGScheduler: Registering RDD 14 (map at <console>:17)
16/04/16 16:51:11 INFO DAGScheduler: Registering RDD 16 (map at <console>:17)
16/04/16 16:51:11 INFO DAGScheduler: Got job 5 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 16:51:11 INFO DAGScheduler: Final stage: ResultStage 10(collect at <console>:22)
```

```
16/04/16 16:51:11 INFO Executor: Finished task 3.0 in stage 10.0 (TID 43). 882 bytes result sent to driver
16/04/16 16:51:11 INFO Executor: Finished task 1.0 in stage 10.0 (TID 41). 882 bytes result sent to driver
16/04/16 16:51:11 INFO Executor: Finished task 2.0 in stage 10.0 (TID 42). 882 bytes result sent to driver
16/04/16 16:51:11 INFO TaskSetManager: Finished task 1.0 in stage 10.0 (TID 41) in 31 ms on localhost (2/4)
16/04/16 16:51:11 INFO TaskSetManager: Finished task 3.0 in stage 10.0 (TID 43) in 31 ms on localhost (3/4)
16/04/16 16:51:11 INFO TaskSetManager: Finished task 2.0 in stage 10.0 (TID 42) in 31 ms on localhost (4/4)
16/04/16 16:51:11 INFO TaskSchedulerImpl: Removed TaskSet 10.0, whose tasks have all completed, from pool
16/04/16 16:51:11 INFO DAGScheduler: ResultStage 10 (collect at <console>:22) finished in 0.004 s
16/04/16 16:51:11 INFO DAGScheduler: Job 5 finished: collect at <console>:22, took 0.151329 s
res5: Array[(String, (Int, Int))] = Array((grapes,(1,1)))
```

```
scala> names1.leftOuterJoin(names2).collect
16/04/16 16:52:16 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 16:52:16 INFO DAGScheduler: Registering RDD 14 (map at <console>:17)
16/04/16 16:52:16 INFO DAGScheduler: Registering RDD 16 (map at <console>:17)
16/04/16 16:52:16 INFO DAGScheduler: Got job 6 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 16:52:16 INFO DAGScheduler: Final stage: ResultStage 13(collect at <console>:22)
16/04/16 16:52:16 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 12, ShuffleMapStage 11)
```

```
16/04/16 16:52:16 INFO Executor: Finished task 1.0 in stage 13.0 (TID 53). 882 bytes result sent to driver
16/04/16 16:52:16 INFO ShuffleBlockFetcherIterator: Getting 3 non-empty blocks out of 4 blocks
16/04/16 16:52:16 INFO ShuffleBlockFetcherIterator: Started 0 remote fetches in 1 ms
16/04/16 16:52:16 INFO TaskSetManager: Finished task 1.0 in stage 13.0 (TID 53) in 38 ms on localhost (3/4)
16/04/16 16:52:16 INFO Executor: Finished task 2.0 in stage 13.0 (TID 54). 1131 bytes result sent to driver
16/04/16 16:52:16 INFO TaskSetManager: Finished task 2.0 in stage 13.0 (TID 54) in 40 ms on localhost (4/4)
16/04/16 16:52:16 INFO DAGScheduler: ResultStage 13 (collect at <console>:22) finished in 0.043 s
16/04/16 16:52:16 INFO TaskSchedulerImpl: Removed TaskSet 13.0, whose tasks have all completed, from pool
16/04/16 16:52:16 INFO DAGScheduler: Job 6 finished: collect at <console>:22, took 0.323516 s
res6: Array[(String, (Int, Option[Int]))] = Array((grapes,(1,Some(1))), (apple,(1,None)), (mango,(1,None)))
```

```
scala> names1.rightOuterJoin(names2).collect
16/04/16 16:53:35 INFO SparkContext: Starting job: collect at <console>:22
16/04/16 16:53:35 INFO DAGScheduler: Registering RDD 14 (map at <console>:17)
16/04/16 16:53:35 INFO DAGScheduler: Registering RDD 16 (map at <console>:17)
16/04/16 16:53:35 INFO DAGScheduler: Got job 7 (collect at <console>:22) with 4 output partitions (allowLocal=false)
16/04/16 16:53:35 INFO DAGScheduler: Final stage: ResultStage 16(collect at <console>:22)
16/04/16 16:53:35 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 15, ShuffleMapStage 14)
16/04/16 16:53:35 INFO DAGScheduler: Missing parents: List(ShuffleMapStage 15, ShuffleMapStage 14)
```

```
16/04/16 16:53:35 INFO TaskSetManager: Finished task 3.0 in stage 16.0 (TID 67) in 23 ms on localhost (2/4)
16/04/16 16:53:35 INFO TaskSetManager: Finished task 1.0 in stage 16.0 (TID 65) in 26 ms on localhost (3/4)
16/04/16 16:53:35 INFO Executor: Finished task 0.0 in stage 16.0 (TID 64). 1115 bytes result sent to driver
16/04/16 16:53:35 INFO TaskSetManager: Finished task 0.0 in stage 16.0 (TID 64) in 32 ms on localhost (4/4)
16/04/16 16:53:35 INFO TaskSchedulerImpl: Removed TaskSet 16.0, whose tasks have all completed, from pool
16/04/16 16:53:35 INFO DAGScheduler: ResultStage 16 (collect at <console>:22) finished in 0.032 s
16/04/16 16:53:35 INFO DAGScheduler: Job 7 finished: collect at <console>:22, took 0.109093 s
res7: Array[(String, (Option[Int], Int))] = Array((grapes,(Some(1),1)), (litchi,(None,1)), (pears,(None,1)))
```

Exercise 4: Operations On Multiple RDDs

Overview

In this lab, we will look at several transformations and examine the optimizations and visualise with DAG.

Builds on

Previous labs for the transformations we'll use.

Run time

30-40 minutes

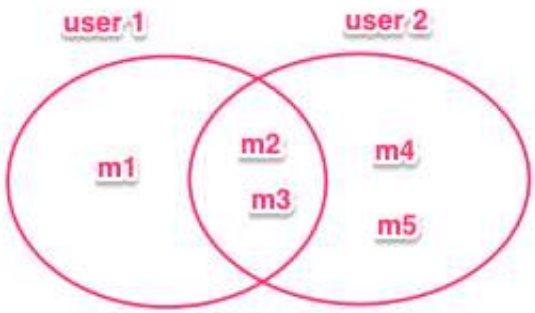
Meetups: Our data for this lab

For this lab, assume that we have users who are attending several meetups. We will start with two users, who are attending the following meetups:

User1 attends meetups: m1, m2 and m3. User2 attends meetups: m2, m3, m4 and m5.

Each user's meetups will be in separate RDDs, so you'll have two RDDs to work with. We'll analyze the data for the two users by performing operations over both RDDs. We'll also look at the Spark Shell UI to get an idea of how Spark is processing the data.

Operations that you work with will include union, intersection, distinct, and subtract .



***** Create the RDDs *****

```
scala> val u1 = sc.parallelize(List("m1", "m2", "m3"))
u1: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[0] at parallelize at <console>:24

scala> val u2 = sc.parallelize(List("m2", "m3", "m4", "m5"))
```

u2: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[1] at parallelize at <console>:24

-- Once you've done that, look at the Jobs tab of the UI - do you see anything?

You won't see any jobs yet. There have been no actions, so Spark is being lazy, and not doing anything yet.

***** Using your two RDDs, find meetups common to both users**

scala> val common = u1.intersection(u2)

common: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[7] at intersection at <console>:28

scala> common.collect

res0: Array[String] = Array(m2, m3)

***** Find meetups attended by either user1 or user2.**

scala> val either = u1.union(u2).distinct()

either: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[11] at distinct at <console>:28

scala> either.collect

res1: Array[String] = Array(m1, m2, m3, m4, m5)

***** Find meetups for each user that only one attended**

-- Find meetups that ONLY u1 attended (That is, u1 attended, but u2 did not.)

scala> val onlyU1 = u1.subtract(u1.intersection(u2))

onlyU1: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[21] at subtract at <console>:28

scala> onlyU1.collect

res2: Array[String] = Array(m1)

-- Find meetups that ONLY u2 attended (That is, u2 attended, but u1 did not.)

```
scala> val onlyU2 = u2.subtract(u1.intersection(u2))

onlyU2: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[31] at subtract at <console>:28


scala> onlyU2.collect

res3: Array[String] = Array(m5, m4)
```

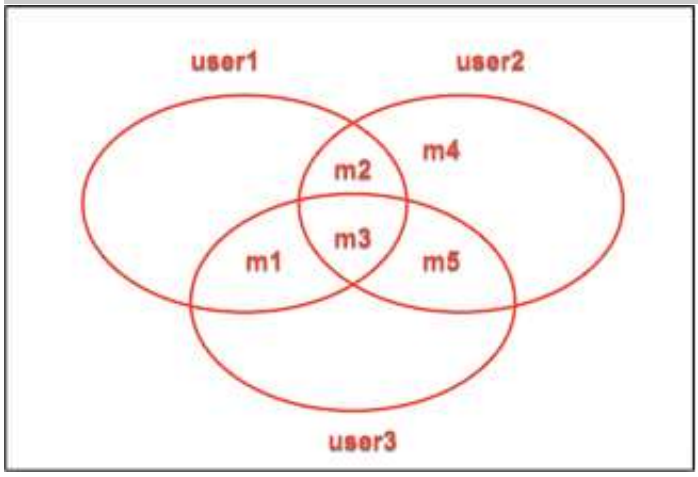
***** Find recommendations based on the requirements in the lab.**

*** A user should not be attending a meetup to be recommended.**

*** A meetup should be in attended by both other users to be recommended.**

Let's introduce user3

User3 attends meetups: m1, m3, and m5, as illustrated below.



Based on this, we would recommend the following to the users:
u1: m5
u2: m1
u3: m2

Tasks

- Create an RDD (u3) with the meetups it is attending (m1, m3, m5).
- Using your three RDDs, find the recommendations for u1 based on the rules above.
  - Look at the diagram above, and put into words how you would compute the recommendations.
  - Once you have a clear idea of how to do it, perform the RDD operations to

accomplish it in the Spark Shell.

- You'll need to use all 3 RDDs in your transformation for this. Look at the job DAG in the UI after you've done this.

-- Create u3

```
scala> val u3 = sc.parallelize(List("m1", "m3", "m5"))

u3: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[32] at parallelize at <console>:24
```

-- Recommendations for u1:

```
scala> val forU1 = u2.intersection(u3).subtract(u1)

forU1: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[42] at subtract at <console>:30

scala> forU1.collect

res4: Array[String] = Array(m5)
```

-- Recommendations for u2:

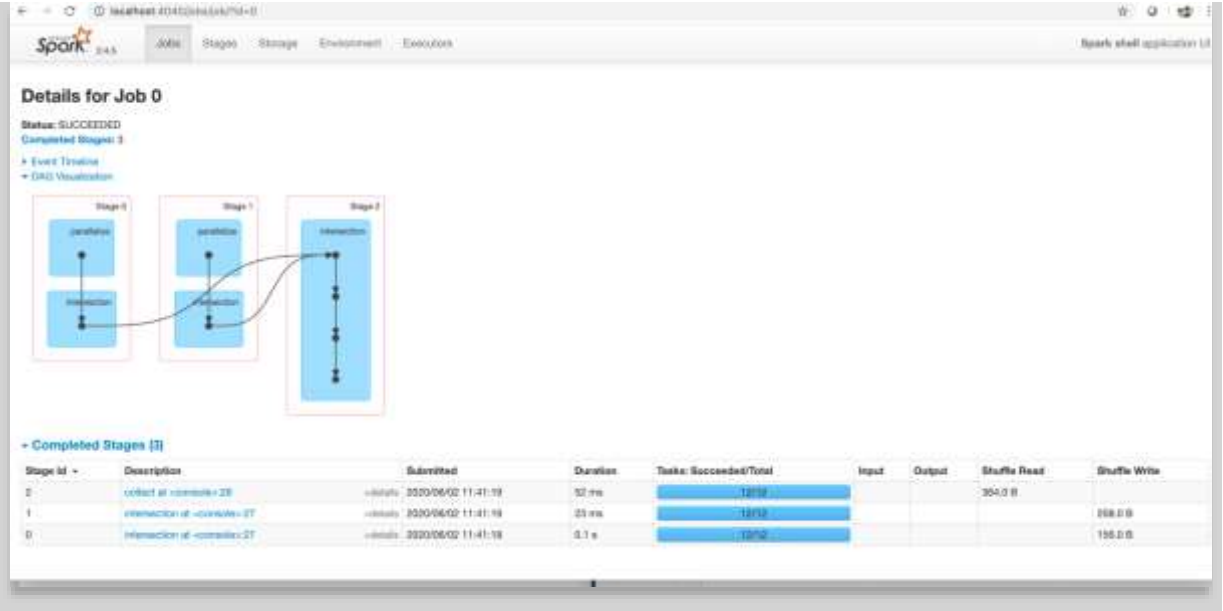
```
scala> val forU2 = u1.intersection(u3).subtract(u2)

forU2: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[52] at subtract at <console>:30

scala> forU2.collect

res5: Array[String] = Array(m1)
```

-- Recommendations for u3:

```
scala> val forU3 = u1.intersection(u2).subtract(u3)

forU3: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[62] at subtract at <console>:30

scala> forU3.collect

res6: Array[String] = Array(m2)
```

Discussions on What's Seen in the Spark Shell UI

Let's consider at some of the results you might have seen earlier in the UI. Here's the DAG from finding the meetups in common (an intersection of the two RDDs).

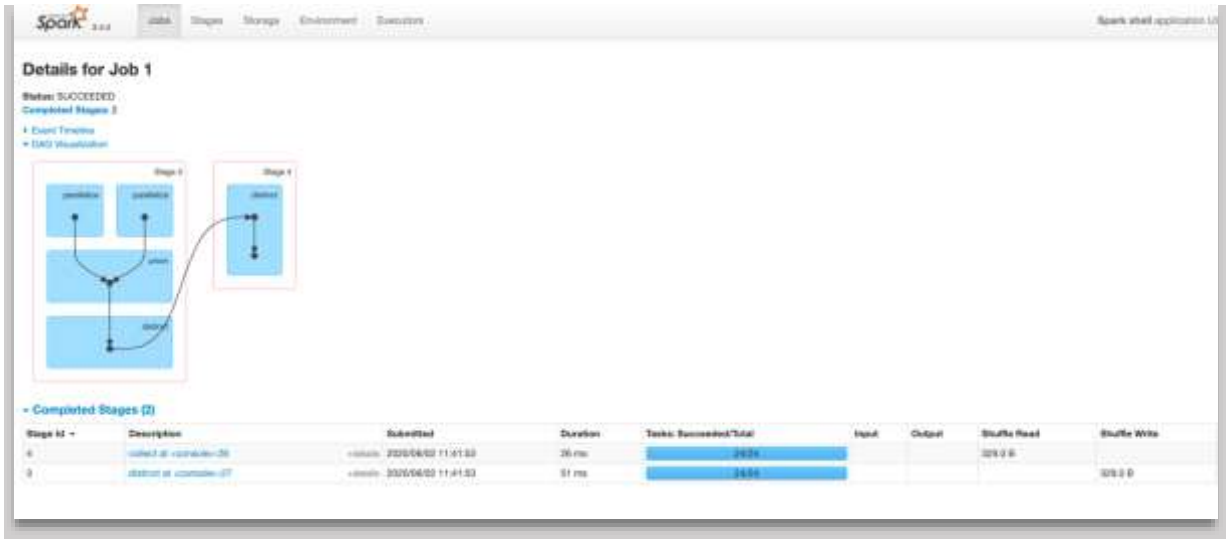


What you're seeing is this.

- Each shaded blue box represents a user operation.
- A dot in the box represents an RDD created in the corresponding operations.
- Operations are grouped by stages (In a stage, operations on partitions are pipelined in the same task).
- You can click on any stage, to drill down into it for more detail.
- Parallelization of each RDD occurs in one stage (e.g. on one node, with local data)
- Some of the intersection can happen on one stage (using whatever data is local)
- Some of the intersection happens in another stage
    - Because it can no longer be done with local data - it involves data distributed over the cluster.
    - Data is **shuffled** for the intersection to be done (i.e. sent from one node to another).
    - Shuffling is expensive - we'll talk more about this later.

Here are the details on the stages (from the **Completed Stages** section on the same page). It gives details on the data that is shuffled.

Here's another DAG - from finding the meetups attended by **either** user (a union of the two RDDs).



What you're seeing is this.

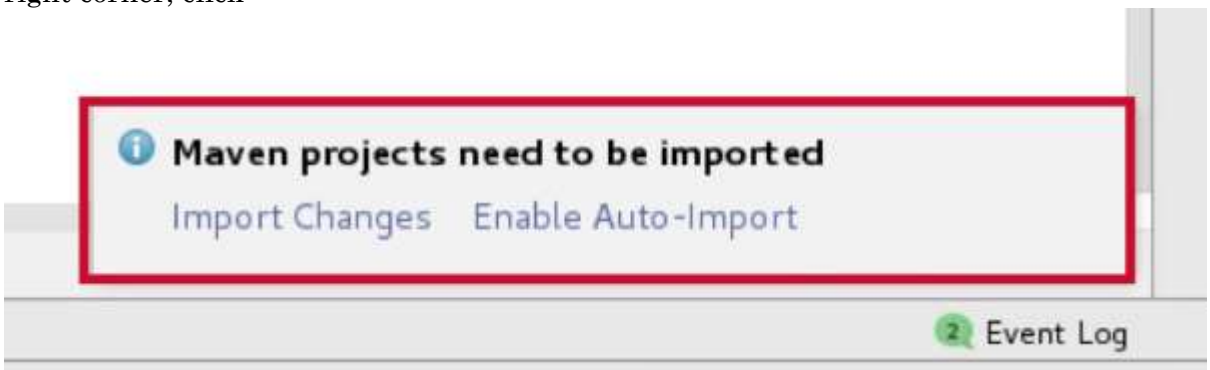
- Parallelization of each RDD occurs in one stage (e.g. on one node, with local data)
  - And yes, Spark has to parallelize again, because default is not to cache an RDD.
- The union happens in the same stage - It can all be done with local data
  - Part of the distinct has to be done in a separate stage - it also requires shuffling data

Exercise5: Building Spark Application using IntelliJ IDE

Overview
In this lab, we will look at several transformations and examine the optimizations that Catalyst performs.
Builds on
Previous labs for the transformations we'll use.
Run time
20-30 minutes

Goal: Create a scala project for Spark. Use maven for lib management.

1. Launch IntelliJ using the link on your lab environment desktop.
2. Click Open to open a project.
3. Locate and select the project's root folder (Lesson4, which you downloaded earlier) and click
4. OK. Wait while the project loads and the workspace is prepared.
5. If there's a "Maven projects need to be imported" popup message in the lower-right corner, click



6. When you have finished the code, build the JAR file which will be submitted to Spark. To do this
   with IntelliJ:
   a. Navigate to View > Tool Windows > Maven Projects. This opens a Maven Projects
   pane on the right side of the screen. You will see sfpdapp listed as a project:
7. Now add a new Scala object file " AirportsByLatitude" to project.

   Use Case: Create a Spark program to read the airport data from airports.text, find all the airports whose latitude are bigger than 40.
   Then output the airport's name and the airport's latitude to airports_by_latitude.
   Each row of the input file contains the following columns:

Airport ID, Name of airport, Main city served by airport, Country where airport is located, IATA/FAA code,
ICAO Code, Latitude, Longitude, Altitude, Timezone, DST, Timezone in Olson format

Sample output:
"St Anthony", 51.391944
"Tofino", 49.082222

```
import org.apache.spark.{SparkConf, SparkContext}


object AirportsByLatitude{
  def main(args: Array[String]) {
    val conf = new SparkConf().setAppName("airports").setMaster("local[2]")
    val sc = new SparkContext(conf)
    val airports = sc.textFile("<hdfs path>/airports.text")
    val airportsInUSA = airports.filter(line =>
line.split(Utils.COMMA_DELIMITER)(6).toFloat > 40)
    val airportsNameAndCityNames = airportsInUSA.map(line => {
      val splits = line.split(Utils.COMMA_DELIMITER)
      splits(1) + ", " + splits(6)
    })
    airportsNameAndCityNames.saveAsTextFile("<hdfs path>/airports_by_latitude.text")
  }
}
```

The code is already imported into IntelliJ in TrainingSpark workspace.

Execute the code, Right click -> Run as Scala Application. It has created the output folder with the desired output. You can go and verify the output.

Use Case:

Create a Spark program to read the airport data from airports.text,
find all the airports which are located in United States and output the airport's name and the city's name to out/airports_in_usa.

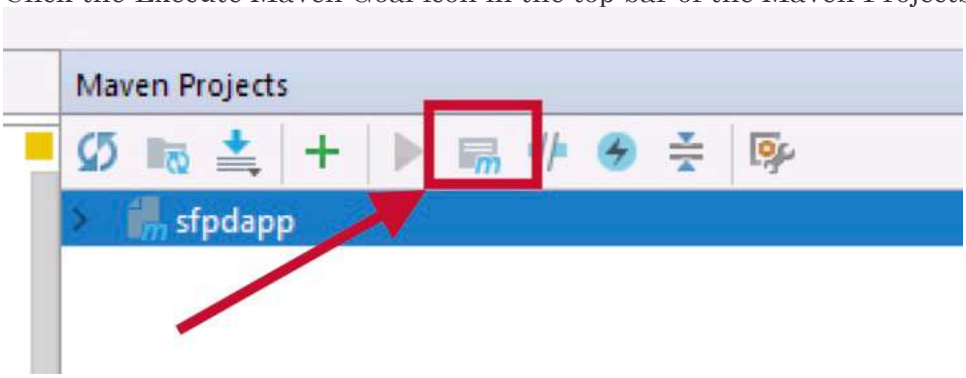
Each row of the input file contains the following columns:

Airport ID, Name of airport, Main city served by airport, Country where airport is located,IATA/FAA code, ICAO Code, Latitude, Longitude, Altitude, Timezone, DST, Timezone in Olson format

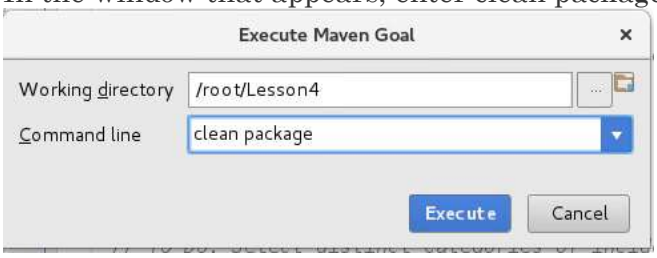
Sample output:

"Putnam County Airport", "Greencastle"
"Dowagiac Municipal Airport", "Dowagiac"

8. Click the Execute Maven Goal icon in the top bar of the Maven Projects window:



In the window that appears, enter clean package, and then click Execute:



9. When it has finished downloading repositories and packaging the project, you will see a BUILD SUCCESS message in the status window. This may take a few minutes.
10. Open a terminal window on the host, and

11. Take this jar and submit using spark-submit command

```
cd /<path>/Lesson4/target

Navigate to the folder where Spark is installed.

$ bin/spark-submit --master yarn --class solution.AirportsByLatitude
/<path>/name of the jar
```

Next verify the output in the output directory of hdfs.

```
(base) [cloudera@quickstart spark-2.4.4-bin-hadoop2.6]$ hdfs dfs -ls /user/cloudera/airports_by_latitude
Found 2 items
-rw-r--r--   3 cloudera cloudera          0 2019-11-16 02:06 /user/cloudera/airports_by_latitude/_SUCCESS
-rw-r--r--   3 cloudera cloudera      92308 2019-11-16 02:06 /user/cloudera/airports_by_latitude/part-00000
(base) [cloudera@quickstart spark-2.4.4-bin-hadoop2.6]$
```

Exercise 6: Accumulators

Here we have the results of stack overflow annual salary survey for developers worldwide. This is a subset of the survey results that's open to CSV via an Excel. The first row is the header we are mostly interested in the third column which is the country of the developer and the 15th column which is the salary made a point.

But as you see we have some responses which don't have the salary midpoint records here. We want to answer several questions.

- First how many records do we have in the survey result.
- Second how many records are missing the salary middle point third.
- How many records are from Canada.

But we want to answer all those questions by passing over all the data only once.

Let us create a Scala class as StackOverFlowSurvey in the package com.sparkTutorial.advanced.accumulator with the below code.

```
package com.sparkTutorial.advanced.accumulator
import com.sparkTutorial.commons.Utils
import org.apache.log4j.{Level, Logger}
```

```
import org.apache.spark.{SparkConf, SparkContext}

object StackOverFlowSurvey {

  def main(args: Array[String]) {
    Logger.getLogger("org").setLevel(Level.ERROR)
    val conf = new
SparkConf().setAppName("StackOverFlowSurvey").setMaster("local[1]")
    val sparkContext = new SparkContext(conf)

    val total = sparkContext.longAccumulator
    val missingSalaryMidPoint = sparkContext.longAccumulator

    val responseRDD = sparkContext.textFile("in/2016-stack-overflow-
survey-responses.csv")

    val responseFromCanada = responseRDD.filter(response => {
      val splits = response.split(Utils.COMMA_DELIMITER, -1)
      total.add(1)

      if (splits(14).isEmpty) {
        missingSalaryMidPoint.add(1)
      }

      splits(2) == "Canada"
    })

    println("Count of responses from Canada: " +
responseFromCanada.count())
    println("Total count of responses: " + total.value)
    println("Count of responses missing salary middle point: " +
missingSalaryMidPoint.value)
  }
}
```