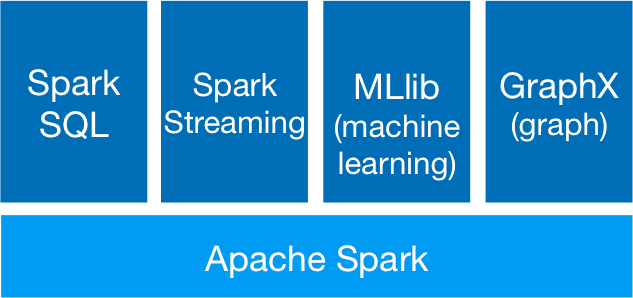
[sparkr개념]

<https://spark.apache.org/docs/latest/rdd-programming-guide.html#rdd-persistence>

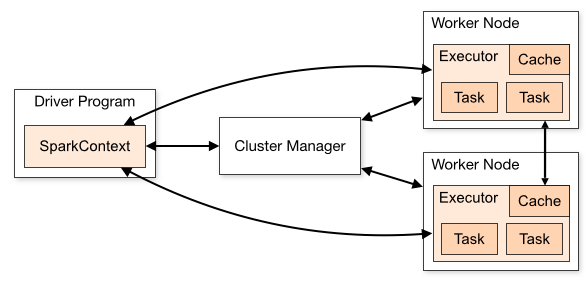
 What is Apache Spark?

* 빠르다
* General Purpose  
  아래 내용들을 하나의 단일 machine 에서 수행가능
  + 배치작업
  + 복잡한 알고리즘(ex. Machine Learning)
  + interactive queries
  + streaming
* Python/Java/Scala/SQL  API 지원
* local filesystem, hdfs, Amazon S3, cassandra, hive, hbase 등 파일 접근 지원

 Spark Stack 구조



* Spark core
  + RDDs
* Spark SQL
  + Hive Table, Parquet, JSON 지원
* Spark streaming
  + 실시간 분석
* MLlib
* GraphX
* Cluster Manager
  + Hadoop Yarn
  + Apache mesos
  + Standalone Scheduler  
    Spark 를 단일 장비에서 수행할 때 기본적으로 설치되는 Scheduler



rdd(resilient distributed dataset): 변경이 불가능한 객체(immutable)의 집합으로 각각의 rdd는 여러개의 파티션으로 분리됨, 변경이 불가능하므로 새로운 rdd를 만들어서 작업

transforamtion: map, filter 등과 같이 즉시 계산이 수행되지 않고 새로운 rdd를 리턴해주는 기록만함

action: count,collect,take 등과 같이 action이 수행될때 실제 계산이 수행됨

map: 각 클러스터간 데이타 교환없이 데이터 변환만 수행하므로 효율적인 병렬처리가능

reduce: 클러스터 내부의 데이터를 reduce계산이 가능하기에 효율적

gorupBy: 데이터를 줄이는 것이 아니라 전부 보전해서 수집해야함

count: rdd의 element 갯수를 세는 동작

collect: rdd의 내용 전부를 드라이버 프로그램으로 가져옴

take(n): 처음 n개의 element를 가져옴

[spark설치]

[root@centos-minion-1 ~]# **wget http://apache.tt.co.kr/spark/spark-2.2.1/spark-2.2.1-bin-hadoop2.7.tgz**

[root@centos-minion-1 **~]# tar xvf spark-2.2.1-bin-hadoop2.7.tgz**

[root@centos-minion-1 bin]# **./spark-shell**

Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties

Setting default log level to "WARN".

To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

17/12/14 23:18:10 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Spark context Web UI available at http://192.168.17.208:4040

Spark context available as 'sc' (master = local[\*], app id = local-1513311492106).

Spark session available as 'spark'.

Welcome to

\_\_\_\_ \_\_

/ \_\_/\_\_ \_\_\_ \_\_\_\_\_/ /\_\_

\_\ \/ \_ \/ \_ `/ \_\_/ '\_/

/\_\_\_/ .\_\_/\\_,\_/\_/ /\_/\\_\ version 2.2.1

/\_/

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0\_151)

Type in expressions to have them evaluated.

Type :help for more information.

**[spark예제]**

scala> **sc.version**

res0: String = 2.2.1

scala> **sc**

res4: org.apache.spark.SparkContext = org.apache.spark.SparkContext@796267e8

scala> **sc.textFile("../README.md")**

res5: org.apache.spark.rdd.RDD[String] = ../README.md MapPartitionsRDD[7] at textFile at <console>:25

scala> **val rdd = sc.textFile("../README.md")**

rdd: org.apache.spark.rdd.RDD[String] = ../README.md MapPartitionsRDD[9] at textFile at <console>:24

scala> **rdd**

res6: org.apache.spark.rdd.RDD[String] = ../README.md MapPartitionsRDD[9] at textFile at <console>:24

scala> **rdd.first**

res7: String = # Apache Spark

scala> **rdd.map(line => line.split(" ") )**

res9: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[10] at map at <console>:27

scala> **res9.take(3)**

res10: Array[Array[String]] = Array(Array(#, Apache, Spark), Array(""), Array(Spark, is, a, fast, and, general, cluster, computing, system, for, Big, Data., It, provides))

scala> **rdd.flatMap(line=> line.split(" "))**

res12: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[11] at flatMap at <console>:27

scala> **res12.take(3)**

res13: Array[String] = Array(#, Apache, Spark)

scala> **rdd.flatMap(line=> line.split(" ")).map(w =>(w,1) )**

res14: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[13] at map at <console>:27

scala> **res14.take(10)**

res15: Array[(String, Int)] = Array((#,1), (Apache,1), (Spark,1), ("",1), (Spark,1), (is,1), (a,1), (fast,1), (and,1), (general,1))

scala> **rdd.flatMap(line=> line.split(" ")).map(w =>(w,1) ).reduceByKey((a,b)=>a+b)**

res19: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[16] at reduceByKey at <console>:27

scala> **res19.take(10)**

res20: Array[(String, Int)] = Array((package,1), (For,3), (Programs,1), (processing.,1), (Because,1), (The,1), (page](http://spark.apache.org/documentation.html).,1), (cluster.,1), (its,1), ([run,1))

scala> **rdd.flatMap(line=> line.split(" ")).map(w =>(w,1) ).reduceByKey((a,b)=>a+b).sortBy(t=>t.\_2).take(10)**

res21: Array[(String, Int)] = Array((package,1), (Programs,1), (processing.,1), (Because,1), (The,1), (page](http://spark.apache.org/documentation.html).,1), (cluster.,1), (its,1), ([run,1), (than,1))

scala> **rdd.flatMap(line=> line.split(" ")).map(w =>(w,1) ).reduceByKey((a,b)=>a+b).sortBy(t=>t.\_2,false).take(5)**

res22: Array[(String, Int)] = Array(("",71), (the,24), (to,17), (Spark,16), (for,12))

scala> **rdd.flatMap(line=> line.split(" ")).map(w =>(w,1) ).countByKey**

res23: scala.collection.Map[String,Long] = Map(site, -> 1, Please -> 4, Contributing -> 1, GraphX -> 1, project. -> 1, "" -> 71, for -> 12, find -> 1, Apache -> 1, package -> 1, Hadoop, -> 2, review -> 1, Once -> 1, For -> 3, name -> 1, this -> 1, protocols -> 1, Hive -> 2, in -> 6, "local[N]" -> 1, MASTER=spark://host:7077 -> 1, have -> 1, your -> 1, are -> 1, is -> 6, HDFS -> 1, Data. -> 1, built -> 1, thread, -> 1, examples -> 2, developing -> 1, using -> 5, system -> 1, than -> 1, Shell -> 2, mesos:// -> 1, 3"](https://cwiki.apache.org/confluence/display/MAVEN/Parallel+builds+in+Maven+3). -> 1, easiest -> 1, This -> 2, -T -> 1, [Apache -> 1, N -> 1, <class> -> 1, different -> 1, "local" -> 1, README -> 1, online -> 1, spark:// -> 1, return -> 2, Note -> 1, if -> 4, project -> 1, Sca...

scala> **val rdd= sc.makeRDD(0 to 1000000000)**

rdd: org.apache.spark.rdd.RDD[Int] = ParallelCollectionRDD[4] at makeRDD at <console>:24

scala> **rdd.count**

res3: Long = 1000000001

scala> **rdd.filter(\_ > 100000)**

res4: org.apache.spark.rdd.RDD[Int] = MapPartitionsRDD[5] at filter at <console>:27

scala> **res4.count**

res5: Long = 999900000

scala> **val rdd = sc.makeRDD( List((1,"A"),(1,"B"),(2,"C"),(2,"D"),(3,"E")))**

rdd: org.apache.spark.rdd.RDD[(Int, String)] = ParallelCollectionRDD[6] at makeRDD at <console>:24

scala> **rdd.groupBy(\_.\_1).collect.foreach(println)**

(1,CompactBuffer((1,A), (1,B)))

(3,CompactBuffer((3,E)))

(2,CompactBuffer((2,C), (2,D)))

scala**> rdd.collect**

res7: Array[(Int, String)] = Array((1,A), (1,B), (2,C), (2,D), (3,E))

scala> **sys.exit() 또는 ctrl+C**

[root@centos-minion-1 bin]# **cd ../sbin**

[root@centos-minion-1 sbin]# **ls -> cluster 구동을 위한 다양한 스크립트 존재**

slaves.sh start-mesos-shuffle-service.sh stop-mesos-dispatcher.sh

spark-config.sh start-shuffle-service.sh stop-mesos-shuffle-service.sh

spark-daemon.sh start-slave.sh stop-shuffle-service.sh

spark-daemons.sh start-slaves.sh stop-slave.sh

start-all.sh start-thriftserver.sh stop-slaves.sh

start-history-server.sh stop-all.sh stop-thriftserver.sh

start-master.sh stop-history-server.sh

start-mesos-dispatcher.sh stop-master.sh

[root@centos-minion-1 sbin]# **./start-all.sh -> 1개 master, 1개 slave 구동**

starting org.apache.spark.deploy.master.Master, logging to /root/spark-2.2.1-bin-hadoop2.7/logs/spark-root-org.apache.spark.deploy.master.Master-1-centos-minion-1.out

root@localhost's password:

localhost: starting org.apache.spark.deploy.worker.Worker, logging to /root/spark-2.2.1-bin-hadoop2.7/logs/spark-root-org.apache.spark.deploy.worker.Worker-1-centos-minion-1.out

[root@centos-minion-1 sbin]# **./stop-all.sh**

root@localhost's password:

localhost: stopping org.apache.spark.deploy.worker.Worker

stopping org.apache.spark.deploy.master.Master

[root@centos-minion-1 spark-2.2.1-bin-hadoop2.7]# **cd conf/**

[root@centos-minion-1 conf]# ls

[root@centos-minion-1 conf]# ll

total 32

-rw-rw-r--. 1 1001 1001 996 Nov 24 18:31 docker.properties.template

-rw-rw-r--. 1 1001 1001 1105 Nov 24 18:31 fairscheduler.xml.template

-rw-rw-r--. 1 1001 1001 2025 Nov 24 18:31 log4j.properties.template

-rw-rw-r--. 1 1001 1001 7313 Nov 24 18:31 metrics.properties.template

-rw-rw-r--. 1 1001 1001 865 Nov 24 18:31 **slaves.template**

-rw-rw-r--. 1 1001 1001 1292 Nov 24 18:31 spark-defaults.conf.template

-rwxrwxr-x. 1 1001 1001 3764 Nov 24 18:31 spark-env.sh.template

[root@centos-minion-1 conf]# **vi slaves.template -> slave에 대한 설정파일**

localhost

**[모니터링 방법]**

master : <http://ip:8080>

worker: http://ip:8081

app: http://ip:4040