免費電子書

學習

apache-spark

Free unaffiliated eBook created from
Stack Overflow contributors.

#apache-spark

1
1: apache-spark 2
2
2
Examples 2
2
3
Spark 4
2: Apache Spark DataFrames 5
Examples 5
JAVASpark DataFrames 5
Spark Dataframe 6
3: Scala 7
7
Examples 7
7
textFile 7
4: Spark DataFrame 8
8
Examples 8
ScalaDataFrame 8
toDF 8
createDataFrame 8
8
5: Spark Launcher 10
10
Examples 10
SparkLauncher 10
6: Spark SQL 11
Examples 11
11

11
11
- 12
7: Spark Streaming 14
Examples 14
PairDStreamFunctions.updateStateByKey 14
PairDStreamFunctions.mapWithState 14
8: 16
Examples 16
16
16
Scala 16
Python 16
9: 18
18
Examples 18
18
RDD 19
RDD 19
19
RDD 20
10: 21
21
Examples 21
Spark 21
11: 24
Examples 24
Scala + JUnit 24
12: SparkJSON 25
Examples 25
GsonJSON 25
13: Apache Spark 26

26

Examples 26

26

26

26

26

26

26

26

27

14: 28

Examples 28

Spark ClientCluster 28

15: pysparkscala 29

29

Examples 29

python RDDScala 29

python RDDscala 29

spark-submit 29

16: Spark 1.6Spark 2.0 30

30

Examples 30

build.sbt 30

ML Vector 30

17: Apache Spark SQL 31

31

Examples 31

Spark SQL Shuffle 31

18: 'sparkR".binsparkR' 32

32

32

Examples 32
Spark for R 32
. 33

1: apache-spark

Apache Spark。/。

---

apache-spark。apache-spark。

2.2.0	2017711
2.1.1	201752
2.1.0	20161228
2.0.1	2016103
2.0.0	2016726
1.6.0	201614
1.5.0	201599
1.4.0	2015611
1.3.0	2015313
1.2.0	
1.1.0	2014911
1.0.0	
0.9.0	201422
0.8.0	2013925
0.7.0	2013227
0.6.0	20121015

Examples

aggregatezeroValueseqOpcombOp

`aggregate()`RDDRDD。

1. `zeroValue` 。
2. `seqOp` RDD。 。

3. combOp ◦

    ◦ (sum, length) ◦

Spark shell42

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

seqOp

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

combOp

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

[1,2]◦ seqOp - (sum, length)◦

local_resultzeroValueaggregate() ◦ 0,0list_element

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

1,111◦ local_result0,01,1◦

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

3,2◦ 7,2◦

combOp

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

'figure'

```
listRDD = sc.parallelize([1,2,3,4], 2)
```

Spark ;◦ ◦

◦

```
In [1]: lines = sc.textFile(file)        // will run instantly, regardless file's size
In [2]: errors = lines.filter(lambda line: line.startsWith("error")) // run instantly
In [3]: errorCount = errors.count()    // an action occurred, let the party start!
Out[3]: 0                              // no line with 'error', in this example
```

[1]SparkRDDlines。“”。

[2]error。 SparkerrorsRDDRDD lineserror。

[3] Spark RDDerrors。 count() Sparkcount()。

[3] [1][2][3]

1. textFile()[1]
2. linesfilter() 'ED[2]
3. count()[3]

Spark[3][1]/[2][3]Spark。 startsWith() [2]Spark[3]Spark[1][2] [2]。

[3][3]

[3] lineserrors。 。 RDDspark。 RDDcache。

Spark/。

Spark

spark-shell

```
sc.version
```

```
sc.version
```

spark-submit

```
sc.version
```

apache-spark https://riptutorial.com/zh-TW/apache-spark/topic/833/apache-spark

2: Apache Spark DataFrames

Examples

JAVASpark DataFrames

DataFrame。 。 DataFrameHiveRDD。

Oracle RDBMSspark::

```
SparkConf sparkConf = new SparkConf().setAppName("SparkConsumer");

sparkConf.registerKryoClasses(new Class<?>[]{
        Class.forName("org.apache.hadoop.io.Text"),
        Class.forName("packageName.className")
});

JavaSparkContext sparkContext=new JavaSparkContext(sparkConf);
SQLContext sqlcontext= new SQLContext(sparkContext);

Map<String, String> options = new HashMap();
options.put("driver", "oracle.jdbc.driver.OracleDriver");
options.put("url", "jdbc:oracle:thin:username/password@host:port:orcl"); //oracle url to
connect
options.put("dbtable", "DbName.tableName");
DataFrame df=sqlcontext.load("jdbc", options);
df.show(); //this will print content into tablular format
```

rdd

```
SparkConf sparkConf = new SparkConf().setAppName("SparkConsumer");

sparkConf.registerKryoClasses(new Class<?>[]{
        Class.forName("org.apache.hadoop.io.Text"),
        Class.forName("packageName.className")
});

JavaSparkContext sparkContext=new JavaSparkContext(sparkConf);
SQLContext sqlcontext= new SQLContext(sparkContext);

Map<String, String> options = new HashMap();
options.put("driver", "oracle.jdbc.driver.OracleDriver");
options.put("url", "jdbc:oracle:thin:username/password@host:port:orcl"); //oracle url to
connect
options.put("dbtable", "DbName.tableName");
DataFrame df=sqlcontext.load("jdbc", options);
df.show(); //this will print content into tablular format
```

```
SparkConf sparkConf = new SparkConf().setAppName("SparkConsumer");

sparkConf.registerKryoClasses(new Class<?>[]{
        Class.forName("org.apache.hadoop.io.Text"),
        Class.forName("packageName.className")
});
```

```
JavaSparkContext sparkContext=new JavaSparkContext(sparkConf);
SQLContext sqlcontext= new SQLContext(sparkContext);

Map<String, String> options = new HashMap();
options.put("driver", "oracle.jdbc.driver.OracleDriver");
options.put("url", "jdbc:oracle:thin:username/password@host:port:orcl"); //oracle url to
connect
options.put("dbtable", "DbName.tableName");
DataFrame df=sqlcontext.load("jdbc", options);
df.show(); //this will print content into tablular format
```

oracle。。rdbms。。/。selectfilteragggroupBy。

Spark Dataframe

SparkDataFrame。R / Python。 DataFrameHiveRDD。

Dataframe

```
val data= spark.read.json("path to json")
```

```
val df = spark.read.format("com.databricks.spark.csv").load("test.txt")val df =
spark.read.format("com.databricks.spark.csv").load("test.txt")
```

RDDDataframe

```
val data= spark.read.json("path to json")
```

df

```
val data= spark.read.json("path to json")
```

SparkRDDDataframe

RDD。

RDDSparkTungsten。RDD。DataframeRDD。

SparkDataFrames。DataFrameCatalystDataFrame。。

DataFrame

Dataframe API。

Apache Spark DataFrames https://riptutorial.com/zh-TW/apache-spark/topic/6514/apache-spark-dataframes

3: Scala

。

Examples

```
val sc: org.apache.spark.SparkContext = ???
sc.textFile(path="/path/to/input/file")
```


```
val sc: org.apache.spark.SparkContext = ???
sc.textFile(path="/path/to/input/file")
```

```
val sc: org.apache.spark.SparkContext = ???
sc.textFile(path="/path/to/input/file")
```


textFile

Spark

- textFile 1

  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


- textFile 2

  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


- 。

  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


  ```
  val txt1=sc.textFile(path="/path/to/input/file1")
  ```


。

Scala https://riptutorial.com/zh-TW/apache-spark/topic/1620/scala

4: Spark DataFrame

DataFrame。SQL。Spark 2.0Dataset [Row]SQL。

Examples

ScalaDataFrame

DataFrame。RDD。

toDF

spark sql implicitsSeqArrayRDDDataFrameProductcaseProduct。

```
import sqlContext.implicits._
val df = Seq(
  (1, "First Value", java.sql.Date.valueOf("2010-01-01")),
  (2, "Second Value", java.sql.Date.valueOf("2010-02-01"))
).toDF("int_column", "string_column", "date_column")
```

createDataFrame

SQLcontext`createDataFrame`。RDD`toDF`。

```
import sqlContext.implicits._
val df = Seq(
  (1, "First Value", java.sql.Date.valueOf("2010-01-01")),
  (2, "Second Value", java.sql.Date.valueOf("2010-02-01"))
).toDF("int_column", "string_column", "date_column")
```

`Row`RDD`schema`DataFrame。

```
import sqlContext.implicits._
val df = Seq(
  (1, "First Value", java.sql.Date.valueOf("2010-01-01")),
  (2, "Second Value", java.sql.Date.valueOf("2010-02-01"))
).toDF("int_column", "string_column", "date_column")
```

DataFrame。hdfs

```
import sqlContext.implicits._
val df = Seq(
  (1, "First Value", java.sql.Date.valueOf("2010-01-01")),
  (2, "Second Value", java.sql.Date.valueOf("2010-02-01"))
```

```
).toDF("int_column", "string_column", "date_column")
```

Spark DataFrame https://riptutorial.com/zh-TW/apache-spark/topic/8358/spark-dataframe

5: Spark Launcher

Spark Launcherspark。 。 ::

```
/** The application has not reported back yet. */
UNKNOWN(false),
/** The application has connected to the handle. */
CONNECTED(false),
/** The application has been submitted to the cluster. */
SUBMITTED(false),
/** The application is running. */
RUNNING(false),
/** The application finished with a successful status. */
FINISHED(true),
/** The application finished with a failed status. */
FAILED(true),
/** The application was killed. */
KILLED(true),
/** The Spark Submit JVM exited with a unknown status. */
LOST(true);
```

Examples

SparkLauncher

spark。 spark。

```
val sparkLauncher = new SparkLauncher
//Set Spark properties.only Basic ones are shown here.It will be overridden if properties are
set in Main class.
sparkLauncher.setSparkHome("/path/to/SPARK_HOME")
  .setAppResource("/path/to/jar/to/be/executed")
  .setMainClass("MainClassName")
  .setMaster("MasterType like yarn or local[*]")
  .setDeployMode("set deploy mode like cluster")
  .setConf("spark.executor.cores","2")

// Lauch spark application
val sparkLauncher1 = sparkLauncher.startApplication()

//get jobId
val jobAppId = sparkLauncher1.getAppId

//Get status of job launched.THis loop will continuely show statuses like RUNNING,SUBMITED
etc.
while (true) {
    println(sparkLauncher1.getState().toString)
}
```

Spark Launcher https://riptutorial.com/zh-TW/apache-spark/topic/8026/spark-launcher

6: Spark SQL

Examples

◦ Spark 1.4◦ UDFsubstrroundSpark 1.4◦ Spark SQL◦ SparkSQLDataFrame API◦

Frame◦ ◦ ◦

-
-
-

- SQL`OVER`avg(revenue) OVER (...);
- DataFrame API`over`rank().over(...) ◦

◦ Spark DataFrame◦

```
val sampleData = Seq(
("bob","Developer",125000),("mark","Developer",108000),("carl","Tester",70000),("peter","Developer",185
```

```
val sampleData = Seq(
("bob","Developer",125000),("mark","Developer",108000),("carl","Tester",70000),("peter","Developer",185
```

`Window Specification` ◦ /◦

```
val sampleData = Seq(
("bob","Developer",125000),("mark","Developer",108000),("carl","Tester",70000),("peter","Developer",185
```

```
val movAvg = sampleData.withColumn("movingAverage", avg(sampleData("Salary"))
             .over( Window.partitionBy("Role").rowsBetween(-1,1)) )
```

- `withColumn()`movingAverageSalaryaverage
- `over()`◦
- `partitionBy()`Role
- `rowsBetween(start, end)`◦ startend 0 -1 1◦ startend-1,0,1◦

```
val movAvg = sampleData.withColumn("movingAverage", avg(sampleData("Salary"))
             .over( Window.partitionBy("Role").rowsBetween(-1,1)) )
```

Spark◦

movingAverage◦ 6◦

```
val cumSum = sampleData.withColumn("cumulativeSum", sum(sampleData("Salary"))
             .over( Window.partitionBy("Role").orderBy("Salary")))
```

- `orderBy()` salary。

```
val cumSum = sampleData.withColumn("cumulativeSum", sum(sampleData("Salary"))
             .over( Window.partitionBy("Role").orderBy("Salary")))
```

-

SparkwithColumnleadlagLevel。 Sparksparksql。SQL。 Spark SQLJSONXMLCSVTSV hetrogenous。

SQLspark SQL。CSV。employeeIDemployeeNamesalarysalaryDate

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

emp.dat。spark CSVspark。

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

myDF。myDF。

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

dataframe

spark。withColumncolumnNameTransformation。empName。

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
```

```
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

“”

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

LAGSQL。。 SparkLAGWindow

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

Lead

LEADSQL。。 SparkLEADWindow

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

LAG。“DOWN”“UP”。WindowWHENif else。

```
1,John,1000,01/01/2016
1,John,2000,02/01/2016
1,John,1000,03/01/2016
1,John,2000,04/01/2016
1,John,3000,05/01/2016
1,John,1000,06/01/2016
```

Spark SQL https://riptutorial.com/zh-TW/apache-spark/topic/3903/spark-sql

7: Spark Streaming

Examples

PairDStreamFunctions.updateStateByKey

updateState by keyDStream ◦

```
object UpdateStateFunctions {
  def updateState(current: Seq[Double], previous: Option[StatCounter]) = {
    previous.map(s => s.merge(current)).orElse(Some(StatCounter(current)))
  }
}
```

currentOptionOption◦

```
object UpdateStateFunctions {
  def updateState(current: Seq[Double], previous: Option[StatCounter]) = {
    previous.map(s => s.merge(current)).orElse(Some(StatCounter(current)))
  }
}
```

PairDStreamFunctions.mapWithState

updateState mapWithStateDStream◦ StateSpec

```
import org.apache.spark.streaming._

object StatefulStats {
  val state = StateSpec.function(
    (key: String, current: Option[Double], state: State[StatCounter]) => {
      (current, state.getOption) match {
        case (Some(x), Some(cnt)) => state.update(cnt.merge(x))
        case (Some(x), None) => state.update(StatCounter(x))
        case (None, None) => state.update(StatCounter())
        case _ =>
      }

      (key, state.get)
    }
  )
}
```

key valueState◦

```
import org.apache.spark.streaming._

object StatefulStats {
  val state = StateSpec.function(
    (key: String, current: Option[Double], state: State[StatCounter]) => {
      (current, state.getOption) match {
```

```
        case (Some(x), Some(cnt)) => state.update(cnt.merge(x))
        case (Some(x), None) => state.update(StatCounter(x))
        case (None, None) => state.update(StatCounter())
        case _ =>
      }

      (key, state.get)
    }
  )
}
```

```
import org.apache.spark.streaming._

object StatefulStats {
  val state = StateSpec.function(
    (key: String, current: Option[Double], state: State[StatCounter]) => {
      (current, state.getOption) match {
        case (Some(x), Some(cnt)) => state.update(cnt.merge(x))
        case (Some(x), None) => state.update(StatCounter(x))
        case (None, None) => state.update(StatCounter())
        case _ =>
      }

      (key, state.get)
    }
  )
}
```

Spark Streaming https://riptutorial.com/zh-TW/apache-spark/topic/1924/spark-streaming

8:

Examples

`SparkContext.broadcast`

```
val broadcastVariable = sc.broadcast(Array(1, 2, 3))
```

`value`

```
val broadcastVariable = sc.broadcast(Array(1, 2, 3))
```

`SparkContext.accumulator`

```
val accumulator = sc.accumulator(0, name = "My accumulator") // name is optional
```

`+=`

```
val accumulator = sc.accumulator(0, name = "My accumulator") // name is optional
```

`value`

```
val accumulator = sc.accumulator(0, name = "My accumulator") // name is optional
```

Spark。。。

---

1. 。value。
2. Java / MapReduce。

Scala

`AccumulatorParam`

```
import org.apache.spark.AccumulatorParam

object StringAccumulator extends AccumulatorParam[String] {
  def zero(s: String): String = s
  def addInPlace(s1: String, s2: String)=  s1 + s2
}
```

```
import org.apache.spark.AccumulatorParam

object StringAccumulator extends AccumulatorParam[String] {
  def zero(s: String): String = s
  def addInPlace(s1: String, s2: String)=  s1 + s2
}
```

Python

`AccumulatorParam`

```
from pyspark import AccumulatorParam

class StringAccumulator(AccumulatorParam):
    def zero(self, s):
        return s
    def addInPlace(self, s1, s2):
        return s1 + s2

accumulator = sc.accumulator("", StringAccumulator())

def add(x):
    global accumulator
    accumulator += x

sc.parallelize(["a", "b", "c"]).foreach(add)
```

https://riptutorial.com/zh-TW/apache-spark/topic/1736/

9:

/。

RDDSpark。 RDD。

。

HadoopHDFSHDFS。

memoryOverhead 。

RDDRDD

```
data = sc.textFile(file)
data = data.coalesce(1)
```

Spark

Spark。

Spark。

RDD

Examples

RDD

HDFS64MB。 。

1. RDDtextFile

   [14]lines = sc.textFile“data”

   [15]lines.getNumPartitionsOut [15]1000

   [16]lines = sc.textFile“data”500

   [17]lines.getNumPartitionsOut [17]1434

   [18]lines = sc.textFile“data”5000

   [19]lines.getNumPartitionsOut [19]5926

[16]RDD。

2. repartition

[22]lines = lines.repartition10

[23]lines.getNumPartitionsOut [23]10

RDD。

Spark。 。

3. coalesce

[25]lines = lines.coalesce2

[26]lines.getNumPartitionsOut [26]2

SparkRDD。 repartitionvs coalesce。

RDD

“”RDD//。 RDD

```
In [1]: mylistRDD = sc.parallelize([1, 2, 3, 4, 5, 6, 7, 8, 9, 10], 2)

In [2]: mylistRDD.getNumPartitions()
Out[2]: 2
```

[1]2`parallelize()`。 RDD2。

RDD

RDD。

```
repartition(numPartitions)
```

```
repartition(numPartitions)
```

RDDRDD。 Spark - repartitionvs coalesce 。

```
repartition(numPartitions)
```

RDD“”100RDD`textFile()`100。

RDDRDD200

```
repartition(numPartitions)
```

RDD34。 。

```
In [1]: data  = sc.textFile(file)

In [2]: total_cores = int(sc._conf.get('spark.executor.instances')) *
int(sc._conf.get('spark.executor.cores'))

In [3]: data = data.coalesce(total_cores * 3)
```

RDD

RDD

```
myRDD.foreach(println)
```

```
myRDD.foreach(println)
```

https://riptutorial.com/zh-TW/apache-spark/topic/5822/

10:

◦ Spark Join◦ Spark◦ 1502◦

Spark-Shell

```
spark-shell   --executor-memory 32G   --num-executors 80  --driver-memory 10g --executor-cores
10
```

Spark

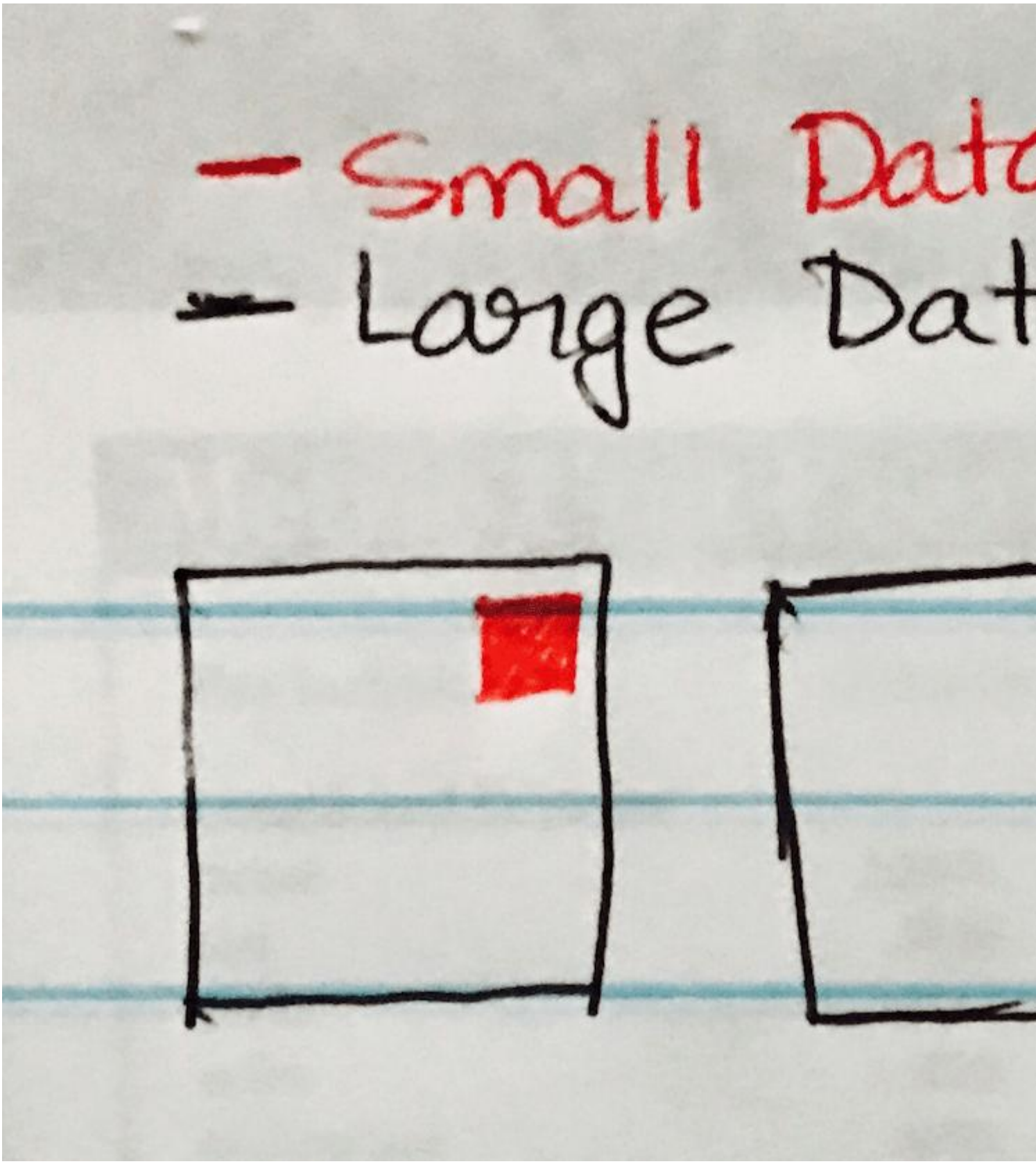
```
spark-shell   --executor-memory 32G   --num-executors 80  --driver-memory 10g --executor-cores
10
```

Examples

Spark

◦ ◦

◦



。

```
case class SmallData(col1: String, col2:String, col3:String, col4:Int, col5:Int)

val small = sc.textFile("/datasource")

val df1 = sm_data.map(_.split("\\|")).map(attr => SmallData(attr(0).toString,
```

```
attr(1).toString, attr(2).toString, attr(3).toInt, attr(4).toInt)).toDF()

val lg_data = sc.textFile("/datasource")

case class LargeData(col1: Int, col2: String, col3: Int)

val LargeDataFrame = lg_data.map(_.split("\\|")).map(attr => LargeData(attr(0).toInt,
attr(2).toString, attr(3).toInt)).toDF()


val joinDF = LargeDataFrame.join(broadcast(smallDataFrame), "key")
```

https://riptutorial.com/zh-TW/apache-spark/topic/7828/

11:

Examples

Scala + JUnit

`countWordsWordCountService`

```
class WordCountService {
    def countWords(url: String): Map[String, Int] = {
        val sparkConf = new
SparkConf().setMaster("spark://somehost:7077").setAppName("WordCount"))
        val sc = new SparkContext(sparkConf)
        val textFile = sc.textFile(url)
        textFile.flatMap(line => line.split(" "))
                .map(word => (word, 1))
                .reduceByKey(_ + _).collect().toMap
    }
}
```

- SparkContext。DI

```
class WordCountService {
    def countWords(url: String): Map[String, Int] = {
        val sparkConf = new
SparkConf().setMaster("spark://somehost:7077").setAppName("WordCount"))
        val sc = new SparkContext(sparkConf)
        val textFile = sc.textFile(url)
        textFile.flatMap(line => line.split(" "))
                .map(word => (word, 1))
                .reduceByKey(_ + _).collect().toMap
    }
}
```

JUnitsparkContextWordCountService

```
class WordCountService {
    def countWords(url: String): Map[String, Int] = {
        val sparkConf = new
SparkConf().setMaster("spark://somehost:7077").setAppName("WordCount"))
        val sc = new SparkContext(sparkConf)
        val textFile = sc.textFile(url)
        textFile.flatMap(line => line.split(" "))
                .map(word => (word, 1))
                .reduceByKey(_ + _).collect().toMap
    }
}
```

https://riptutorial.com/zh-TW/apache-spark/topic/3333/

12: SparkJSON

Examples

GsonJSON

Gson JSONMyClass ◦

GsonGson◦ MyClass◦

jsonJSON◦ JSON◦ JSON◦

```
val sc: org.apache.spark.SparkContext // An existing SparkContext

// A JSON dataset is pointed to by path.
// The path can be either a single text file or a directory storing text files.
val path = "path/to/my_class.json"
val linesRdd: RDD[String] = sc.textFile(path)

// Mapping json to MyClass
val myClassRdd: RDD[MyClass] = linesRdd.map{ l =>
    val gson = new com.google.gson.Gson()
    gson.fromJson(l, classOf[MyClass])
}
```

GsonmapPartitions◦ Gson

```
val sc: org.apache.spark.SparkContext // An existing SparkContext

// A JSON dataset is pointed to by path.
// The path can be either a single text file or a directory storing text files.
val path = "path/to/my_class.json"
val linesRdd: RDD[String] = sc.textFile(path)

// Mapping json to MyClass
val myClassRdd: RDD[MyClass] = linesRdd.map{ l =>
    val gson = new com.google.gson.Gson()
    gson.fromJson(l, classOf[MyClass])
}
```

SparkJSON https://riptutorial.com/zh-TW/apache-spark/topic/2799/sparkjson

13: Apache Spark

Apache Spark。

Examples

Apache Spark

- Apache SparkSpark。 API1.6,2.0,2.1。
- ScalaSpark。
- JDK `java -version` 。
- PythonR。 Python python-2.x python-3.x。
- `build.sbt` `pom.xml` PythonR。
- `local[n]` Spark standaloneYarnMesos `client` `cluster` 。

。

- RDDAPI。
- API`StrucTypeDataset.printSchema` 。

`Dataset.showprint`。

Spark`org.apache.spark.mllib.random.RandomRDDsorg.apache.spark.graphx.util.GraphGenerators`。

。 。

```
val lines: RDD[String] = rdd.map(someFunction)
```

```
val lines: RDD[String] = rdd.map(someFunction)
```

```
val lines: RDD[String] = rdd.map(someFunction)
```

```
val lines: RDD[String] = rdd.map(someFunction)
```

。

。

。 。

。

- `RDD.debugString` / `Dataset.explain` 。
-

DAGSpark UI。

- 。
- GangliaVisualVM。

- 。 。
- 。
-
- Apache Spark

Apache Spark https://riptutorial.com/zh-TW/apache-spark/topic/8815/apache-spark-

14:

Examples

Spark ClientCluster

Spark。

SparkSparkContextDriver;。 。 Spark。 。 。 “”。 。

Worker。 Worker。 JAR。 。 。 JVM。

Apache Mesos - Hadoop MapReduce。 Hadoop YARN - Hadoop。
Kubernetes-infrastructure.it。

https://riptutorial.com/zh-TW/apache-spark/topic/10808/

15: pysparkscala

pysparkScala。

Python APIScala APIpython。

Python`scikit-learn`。

Examples

python RDDScala

python RDDScala。 JavaRDD [Any]

```
import org.apache.spark.api.java.JavaRDD

def doSomethingByPythonRDD(rdd :JavaRDD[Any]) = {
    //do something
    rdd.map { x => ??? }
}
```

python RDDscala

python RDDJVM。 Sparkjar。

```
from pyspark.serializers import PickleSerializer, AutoBatchedSerializer


rdd = sc.parallelize(range(10000))
reserialized_rdd = rdd._reserialize(AutoBatchedSerializer(PickleSerializer()))
rdd_java = rdd.ctx._jvm.SerDe.pythonToJava(rdd._jrdd, True)

_jvm = sc._jvm #This will call the py4j gateway to the JVM.
_jvm.myclass.apps.etc.doSomethingByPythonRDD(rdd_java)
```

spark-submit

scalajar。

```
spark-submit --master yarn-client --jars ./my-scala-code.jar --driver-class-path ./my-scala-
code.jar main.py
```

pySparkscala

pysparkscala https://riptutorial.com/zh-TW/apache-spark/topic/9180/pysparkscala

16: Spark 1.6Spark 2.0

Spark 2.0。Spark 1.6Spark 2.0API。。

Examples

build.sbt

build.sbt

```
scalaVersion := "2.11.8" // Make sure to have installed Scala 11
sparkVersion := "2.0.0"  // Make sure to have installed Spark 2.0
```

`sbt package .jartarget/scala-2.11/.jarspark-submit。`

ML Vector

ML `Transformersorg.apache.spark.ml.linalg.VectorUDTorg.apache.spark.mllib.linalg.VectorUDT。`

`org.apache.spark.ml.linalg.Vector。` MLLib APISpark 2.0.0。

```
//import org.apache.spark.mllib.linalg.{Vector, Vectors} // Depreciated in Spark 2.0
import org.apache.spark.ml.linalg.Vector // Use instead
```

Spark 1.6Spark 2.0 https://riptutorial.com/zh-TW/apache-spark/topic/6506/spark-1-6spark-2-0

17: Apache Spark SQL

SparkSpark SQLApache Spark。

Examples

Spark SQL Shuffle

Apache Spark`joincogroup`。 shuffleSpark SQL。

```
spark.sql.shuffle.partitions
```

shuffle。 `200` 。 GB200。 Spark SQL。

`employeedepartment` 。 。 Spark DataFrames

```
spark.sql.shuffle.partitions
```

200。

shuffle。

```
spark.sql.shuffle.partitions
```

shuffle2DataFrames`0.878505 s0.077847 s` 。

。

Apache Spark SQL https://riptutorial.com/zh-TW/apache-spark/topic/8169/-apache-spark-sql

18: 'sparkR''.binsparkR'

WindowsSpark。sparkRR。

r-bloggers

Examples

Spark for R

URL - https://www.r-bloggers.com/installing-and-starting-sparkr-locally-on-windows-os-and-rstudio-2/'Spark / bin''spark / bin'RRstudio。C\ spark-2.0.1 C\ spark-2.0.1 \ bin C\ spark-2.0.1 \ sbin C\ Program Files \ R \ R-R / 3.3.1 \ bin \ x64 C\ Program Files \ RStudio \ bin \ x64

Windows 10Windows 8“”“”。Sytem PropertiesAdvancedEnvironment Variables。“”PATH。。PATH“”。“”“”PATH。。“”。sparkR。

Windows 7“”。“”。“”。“”。“”PATH。。PATH“”。“”“”PATH。。“”。sparkR。

'sparkR''.binsparkR' https://riptutorial.com/zh-TW/apache-spark/topic/9649/-sparkr---binsparkr-

S. No		Contributors
1	apache-spark	4444, Ani Menon, Community, Daniel de Paula, David, gsamaras, himanshuIIITian, Jacek Laskowski, KartikKannapur, Naresh Kumar, user8371915, zero323
2	Apache Spark DataFrames	Mandeep Lohan, Nayan Sharma
3	Scala	Ani Menon, Community, spiffman
4	Spark DataFrame	Daniel de Paula
5	Spark Launcher	Ankit Agrahari
6	Spark SQL	Daniel Argüelles, Hari, Joshua Weinstein, Tejus Prasad, vdep
7	Spark Streaming	zero323
8		Community, Jonathan Taws, RBanerjee, saranvisa, spiffman, whaleberg, zero323
9		Ani Menon, Armin Braun, gsamaras
10		Adnan, CGritton
11		Cortwave
12	SparkJSON	Furkan Varol, zero323
13	Apache Spark	user7337271
14		Nayan Sharma
15	pysparkscala	eliasah, Thiago Baldim
16	Spark 1.6Spark 2.0	Béatrice Moissinac, eliasah, Shaido
17	Apache Spark SQL	himanshuIIITian
18	'sparkR''.binsparkR'	Rajesh